Brigham Young University

AUVSI-SUAS Competition Team 2019

BYU Unmanned Aircraft System Technical Design

Abstract

The Brigham Young University AUVSI-SUAS competition team designed an unmanned aircraft system capable of completing all mission requirements for the AUVSI-SUAS 2019 competition. The UAS was designed and built by 12 students from the departments of mechanical, electrical, and computer engineering with the aim to maximize point potential while operating in a safe and repeatable manner. Extensive flight testing has validated the design, consisting of a custom-built autopilot, a modified Nimbus Pro airframe, an imaging system, and an unmanned ground vehicle payload delivery system.

1 Systems Engineering Approach

Our team's unmanned aircraft system (UAS) was designed, built, and tested with the aim to maximize our score in the AUVSI SUAS competition while minimizing safety risks and complying with relevant AMA guidelines. The integrated system is modular and has been validated with both simulation and hardware flight testing. The following sections demonstrate how the UAS is capable of performing all mission tasks in a reliable and safe manner. Mission requirements are discussed as well as how those requirements translated into our design rationale and testing procedures for each of the major UAS subsystems.

1.1 Mission Requirement Analysis

Requirements placed on the UAS include the ability to fly autonomously; navigate waypoints; avoid obstacles; detect, classify, and geolocate ground targets; and deliver a smart payload, all within the time limit and in a professional manner. Table 1 details each requirement with its associated scoring weight. A third column in the table briefly describes the system to be constructed to meet the associated requirement.

Effective design must consider tradeoffs. For example, the airframe must have sufficient stability, range, and speed, all of which come at the expense of maneuverability (important for navigation). When designing the autopilot, robustness is generally obtained at the expense of algorithm speed, which can also affect navigational precision. The reliability of the network connection can be improved by using high-power hardware, which can increase weight and cost. Image resolution can be increased at the expense of required stream time. It can also be improved by decreasing flight speed, which inherently affects timeline. Greater functionality can be added to the payload drop system at the expense of weight and complexity. Finally, all software and hardware development can be improved at the opportunity cost of improving other systems. These should be coordinated to maximize the scoring functions included in Table 1.

1.2 Design Rationale

Our system design considers both environmental factors and mission requirements. Environmental factors included our team resources including $3,500 (travel excluded), 2,500 man hours, current team-member expertise, and resources carried-over from last year's team. Decisions were made that would build on last year's design, stay within our budget, and play to our strengths as a team, which consists of seniors majoring in mechanical, electrical, or computer engineering. The competition was also valued as a learning opportunity. As such, some decisions were made based on what we could be learned from them, even when it meant taking a more difficult approach. In addition to these factors, maximizing points scored during the mission portion of the competition was emphasized. Most of our efforts focused on autonomous flight, obstacle avoidance, air drop, object detection and classification, and required communications.

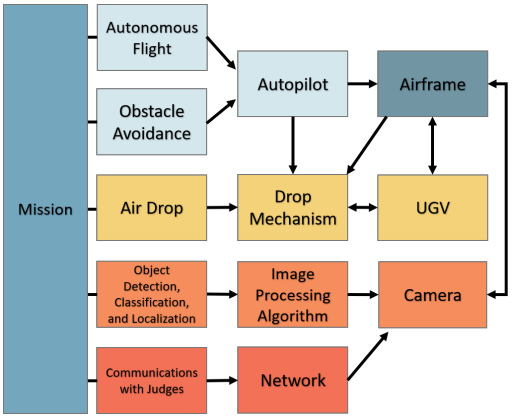

Figure 1: System design decision tree.

Our design decision flow is shown in Figure 1. Our first consideration was the chosen autopilot system to successfully complete the autonomous flight and obstacle avoidance portions of the mission. This led to our decision to use a fixed-wing airframe design. The payload drop and unmanned ground vehicle (UGV) decisions were made to maximize mission points with additional consideration to integrating with the airframe—the UGV also needed to be light-weight and small enough to fit inside our potential airframe. Image detection and networking systems were chosen based on the mission re-

Table 1: High-level mission definition, including categories, weights, specific requirements for the UAS, and a brief subsystem description.

Category (Weight)	Requirement	Subsystem Description
Timeline (10%)		
Mission Time (80%)	- Fly up to 4 miles of waypoints, image roughly 250,000 m^2, stream images, and drop a payload within 40 minutes.	- An airframe with sufficient endurance, speed, and maneuverability with fast network connection.
Timeout (20%)	- Execute the mission without needing a timeout.	- Reliable network hardware, robust autopilot software.
Autonomous Flight (20%)	- Avoid manual takeovers, items falling off the plane, and crashing.	
Autonomous Flight (40%)	- Fly autonomously for at least 3 minutes.	- Robust autopilot software and airframe.
Waypoint Capture (10%)	- Fly within 100 ft of up to 4 miles of waypoints while remaining in bounds.	- Airframe with sufficient range.
Waypoint Accuracy (50%)	- Fly as close as possible to each waypoint.	- Precise global positioning system (GPS).
Obstacle Avoidance (20%)	- Avoid static obstacles and upload telemetry at 1 Hz.	- Path-planner with obstacle avoidance. Ground station link uploads telemetry.
Object Classification (20%)		
Characteristics (20%)	- Recognize target characteristics.	- Software crops and identifies images.
Geolocation (30%)	- Provide GPS coordinates of targets.	- Software geolocates targets.
Actionable (30%)	- Submit objects while still airborne.	- Imaging system links with ground station during flight.
Autonomy (20%)	- Submit objects autonomously.	- Imaging system maximally autonomous.
Air Drop (20%)		
Drop Accuracy (50%)	- Gently deliver a UGV and water bottle to provided GPS coordinates.	- Precise and gentle drop mechanism.
Drive to Location (50%)	- Drive within 10 ft of destination while remaining in bounds.	- UGV lands in a drivable state and has a sufficiently precise autopilot and GPS.
Operational Excellence (10%)	- Demonstrate professionalism, effective communication, safety, etc.	- Pre-flight checklist documents potential problems and solutions. Team developed habits of professionalism and safety.

quirements, which influenced our camera decision. The camera also had a dual-dependency with the airframe decision, as the camera was chosen to be light-weight while fitting in the airframe, and the airframe was chosen to fly slower than other options to improve image quality. The final airframe design was then chosen to ensure these components could fit inside and were adequately protected.

1.2.1 Autopilot subsystem

A common approach to the AUVSI-SUAS competition is a Pixhawk-based system. However, off-the-shelf autopilots like Pixhawk can be difficult to adapt programmatically to the mission requirements. Instead, a Robot Operating System (ROS)-based autopilot called ROSplane (https://github.com/BYU-AUVSI/rosplane), developed here at BYU, was selected. ROSplane utilizes deep integration with ROS to enable platform flexibility. The open-source and robust nature of ROSplane means that algorithm customization and optimization is relatively easy. This includes total customization of the control and path planning scheme.

ROSplane also uses the Gazebo simulation development environment to enable high-fidelity simulations that reduce dependency on in-air testing time. The accompanying flight controller firmware, ROSflight (https://github.com/BYU-AUVSI/rosflight), also provides a platform that allows customization all the way down to the pulse width modulation (PWM) output to servos.

1.2.2 Airframe subsystem

Eight different designs were considered for the airframe, including custom fixed-wing or flying-wing

designs, a hexacopter, and various off-the-shelf fixed-wing designs. Factors in our decision included lift, flight speed, internal payload volume, stability control, cost, range, estimates on time to assemble/rebuild, and estimates on time required to integrate with the autopilot. Based on all of these considerations, the Nimbus Pro airframe by My Fly Dream was chosen as our airframe design. This off-the-shelf fixed-wing design is quick to assemble and rebuild—an important feature, as BYU's team crashed just days before last year's competition. It also has a larger wing area than our other design options. This decreases stall speed and in turn allows for clearer images and easier flight navigation. The Nimbus Pro also has ample room to house the camera and UGV. Fully loaded, it even has extra room to adjust the center of gravity as necessary by moving components.

1.2.3 Imaging subsystem

Factors in selecting the imaging system included weight, size, resolution, stabilization, and ability to stream images over a network. Many of these factors were strongly influenced by the choice of airframe. With the Nimbus Pro, abundant space was available for a larger imaging system then the one used last year. With this in mind, the Sony a6000 was selected. This consumer grade camera is inexpensive, compact and can be interfaced with a computer to automatically stream images over a network. The a6000 also has excellent resolution and image stabilization, allowing targets to be easily resolved both manually and autonomously at flight altitudes.

1.2.4 Air drop payload subsystem

To minimize aerodynamic drag and to prevent interference with other flight operations (e.g. takeoff and emergency landing), the UGV was designed to be carried inside the fuselage until dropped. As such, the inner dimensions of the airframe constrained the size of the UGV to no larger than $15 \times 15 \times 20$ cm. Designing a small and light UGV also became important to reduce the impact on center of gravity after delivery, and a total mass constraint of 0.5 kg was imposed. The final design chosen to meet these requirements is a chassis from a small remote control car, modified to respond to PWM commands from a microcontroller. Because components further from the center of gravity experience greater vibrations, it was postulated that placing the camera near the center of gravity (CG) would increase image quality. As such, the UGV and associated dropping mechanism were placed slightly aft of the aircraft's CG.

Of note, is that mission requirements call for a gentle landing of the UGV. In testing, it was determined that a landing velocity less than 3 m/s is gentle enough to prevent damage to our ground vehicle. This maximum value was used to validate different payload drop concepts. Ultimately, it was determined that a parachute provides the best combination of ease of development, low landing velocity, and predictable landing location.

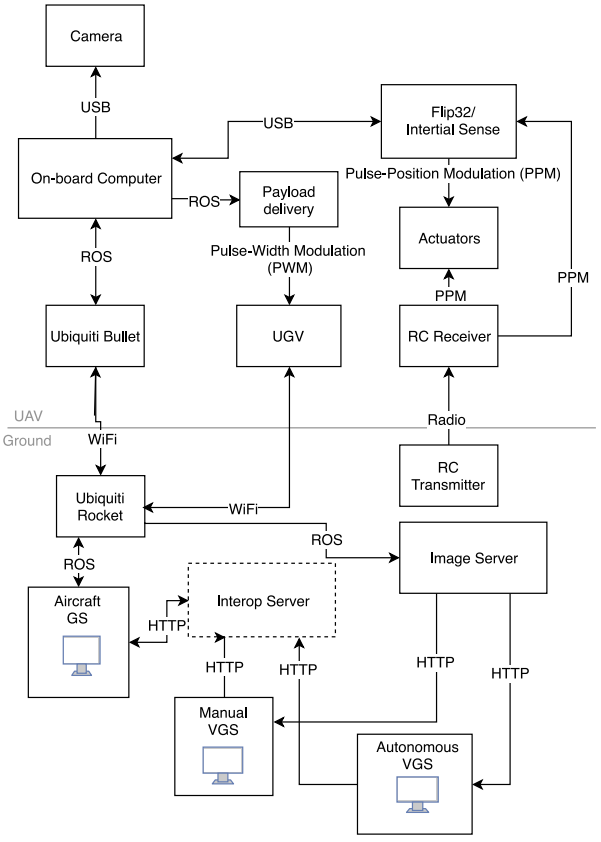

Figure 2: System-wide interface diagram for the UAS.

2 System Design

The UAS is a complex system of interconnected components, each with their own design rationale and testing procedures. Figure 2 gives a top-level description of all major components on the ground and in the air, as well as their communication interfaces.

The following sections detail the design and testing process for each major component of the UAS. Each component was designed to meet the design requirements in Table 1.

2.1 Aircraft

To provide a modular design that can be rebuilt quickly in the case of a catastrophic crash, an off-the-shelf airframe—My Fly Dream's Nimbus Pro (Fig. 3 and 4—also see Table 2 for airframe specifications)—was selected. This airframe boasts a sturdy, expanded, polystyrene body with two carbon-fiber spars in the main wings and a single carbon-fiber spar in the tail and fin. A 1/8 in wood sheet fortifies the front of the fuselage and plastic clips facilitate easy attachment and removal of the wings and tail. A single electrical connector provides power, motor PWM signal, and control surface PWM signal to the wings and facilitates quick wing attachment.

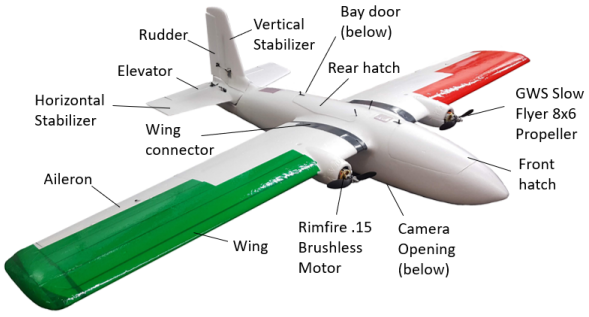

Figure 3: Labeled components of the Nimbus Pro airframe.

To optimize aerodynamic efficiency, the airframe was analyzed using the aerodynamics software XFLR5. Drag polars obtained during analysis are included in Figure 5. Based on this simulation, a target design angle of attack of 7 deg was selected

Table 2: Physical properties and specifications of the aircraft

Airframe Specifications	
Model	Nimbus Pro by My Fly Dream
Material	Expanded Polystyrene
Wing Span	1.9 m
Wing Area	0.473 m^2
Length	1.283 m
Gross Weight	5.2 kg
Tail Incidence	7.5 deg
Motor (× 2)	Rimfire .15 1200 kV
Propeller (× 2)	GWS Slow Flyer 8×6

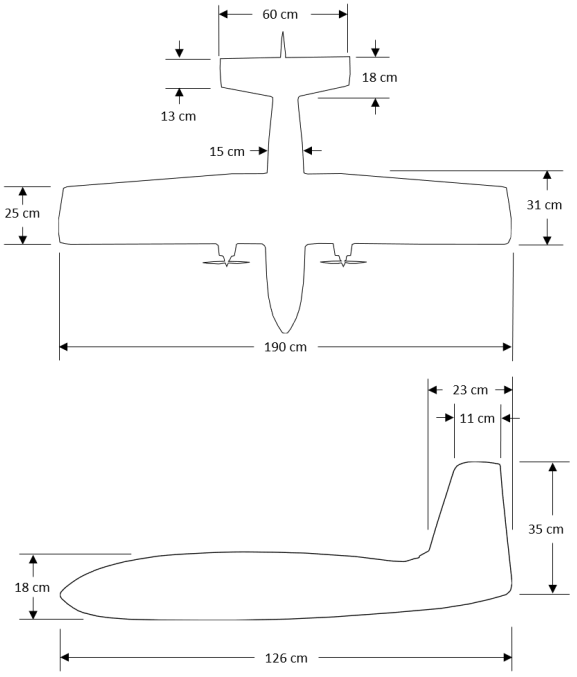

Figure 4: Dimensioned top and side views of the aircraft.

to achieve the optimal lift-to-drag ratio (L/D) while avoiding stall.

To fly at the design angle of attack, the CG and tail incidence angle were adjusted. The tail incidence angle of the Nimbus Pro is zero, requiring significant elevator deflection for steady level flight, increasing the required elevator trim and causing unnecessary drag. To remedy this, the airframe's tail-incidence angle was modified. In XFLR5, the tail incidence angle and CG were adjusted iteratively. Figure 5

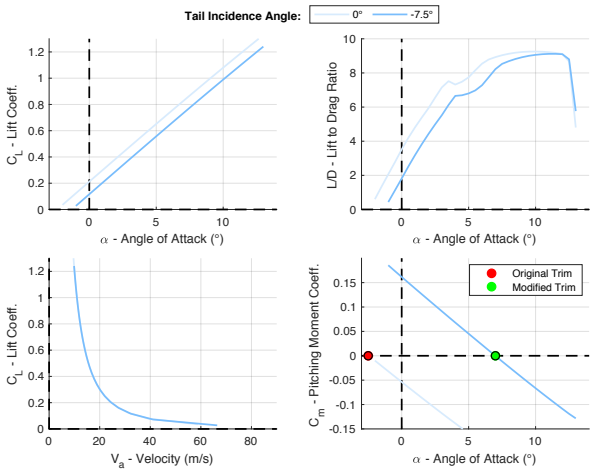

Figure 5: XFLR5 analysis of the airframe at two different tail incidence angles. As can be seen, decreasing the tail incidence angle from 0.0 deg to -7.5 deg trims the aircraft near our maximum lift-to-drag ratio.

shows the aerodynamic performance of the original and adjusted tail incidence angles. The effect of the tail incidence angle is most apparent in the $C_{m,\alpha}$ plot (bottom right). In this figure, note that the 0 deg tail incidence configuration is unacceptable without elevator deflection, as it would fly at a negative angle of attack, resulting in a near-zero lift coefficient. Contrast this with the blue curve of the $C_{m,\alpha}$ plot, which shows statically stable flight at an angle of attack of our target 7 deg.

Next, CG location was considered. Since steady level flight occurs when the sum of longitudinal pitching moments equals 0 (i.e. $C_m = 0$ on the C_m vs. α plot), adjusting the lever arm of the plane's weight has a strong influence on the stable angle of attack. After iterating on the CG and tail incidence angle to maximize the lift-to-drag ratio (shown on the C_L vs. C_D plot) and minimize design velocity (shown on the C_L vs. V plot), it was determined that a CG placement of 6.2 cm aft of the leading edge of the wing is optimal. The resulting tail incidence angle was 7.5 deg. These two factors combined to achieve a design angle of attack of about 7 deg, consistent with our target value. We calculated the static margin to be about 5%. Though slightly low, the increased aerodynamic performance was determined to be worth the small static margin. Components of the system were placed inside the airframe to achieve the desired aircraft CG placement (Fig. 6). Also of note, the airframe was modified using an extruded polystyrene foam wedge to increase tail incidence to 7.5 deg. When the resulting configuration was built and tested in a field test, almost no trim was required for steady level flight, validating the concept.

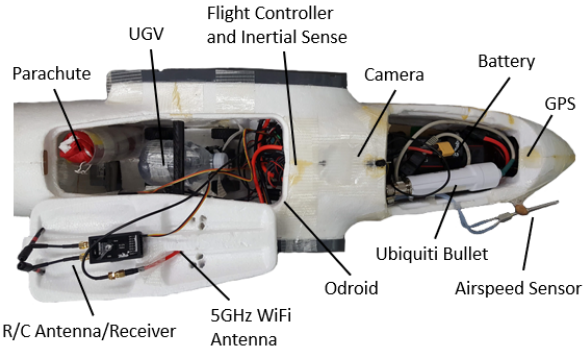

Figure 6: Layout of components of the UAS.

For propulsion, two Rimfire 1200 kV brushless electric motors were selected with a 4-cell lithium-polymer battery, based on their excellent performance last year. Using propeller performance data published by the University of Illinois Urbana-Champaign, 56 propellers of various sizes and geometries were modeled for efficiency at our design speed and with our selected motors and battery. Of the propellers available, GWS Slowflier 8×6 propellers were selected with an efficiency of 40.5%, an improvement over last year's efficiency of 35.5%. Selected components are summarized in Table 2.

To select the gauge of the wire used, max current to the motors was measured empirically using a current clamp meter with the plane held stationary and maximum throttle applied. A maximum current of 35 A was measured, justifying the use of 12 gauge wire for the motors with a 10 gauge wire power harness from the battery. A large safety factor of two was desired for the ESC's to prevent overheating, and since no significant weight disparity was found between 70 A and 100 A ESC's available online, 100 A ESC's were chosen.

The performance of the aircraft (see Table 3) was

estimated using XFLR5 and then verified over the course of 31 flight tests totalling 3.8 hours of flight. The cruise speed was found to be 17.4 m/s with a lift over drag ratio of 10. The total range of 52 km was estimated using the measured battery endurance and cruise speed.

Table 3: Measured performance of the aircraft. Range is extrapolated from endurance and cruise speed.

Airframe Performance	
Cruise Speed	17.4 m/s
Stall Speed	12 m/s
Range	52 km
Endurance	50 min
Motor/Prop Efficiency	40.49%
Total # of Flights	31
Autonomous Flight Time	40 min
Total Flight Time	3.83 hr

2.2 Autopilot

The ROSplane autopilot consists of various nodes, seen in Figure 7, that perform specialized tasks and communicate over ROS messages.

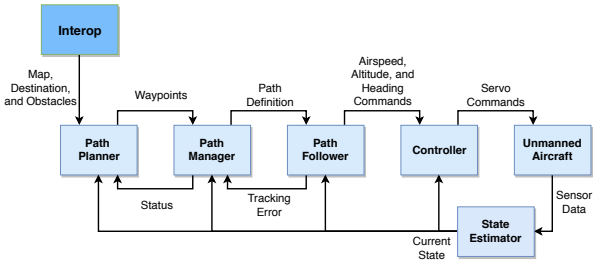

Figure 7: A block diagram showing the various nodes within the autopilot. Beard and McLain. "Small Unmanned Aircraft," Princeton University Press, 2012.

The path manager receives a series of desired waypoints that need to be flown through. For each pair of consecutive waypoints, a path is defined by a straight line and then modified with circles to provide a smooth transition between line segments. The circles are defined by the minimum turning radius of the UAS to ensure that the path is flyable and the distance to the waypoint is minimized. The path manager tracks the position of the UAS, and once the UAS completes each individual segment, the path manager passes the next path segment to the path follower.

The path follower takes in a path definition that describes either a straight line or a circle. A straight line is defined by an origin point and a unit vector defining a direction. A circle is defined by the center of the circle, a radius, and a direction of travel. The path follower determines the necessary course heading and altitude needed to minimize the perpendicular distance between the defined path and the UAS position. The desired airspeed is determined by the specific task the UAS is attempting to accomplish. These commands are then passed to the controller.

The controller converts desired airspeed, altitude, and heading commands into control surface deflections using a series of proportional-integral-derivative (PID) controllers. The airspeed is controlled by varying the throttle command using a proportional-integral (PI) controller. The desired altitude sets a desired pitch using a PI controller and then a proportional-derivative (PD) controller determines the necessary elevator deflection to obtain that pitch. Finally, the desired course uses a PI controller to determine the necessary roll and a PD controller uses that roll to determine aileron deflections.

The UAS, described in Section 2.1, receives pulse width modulation (PWM) commands from the controller that dictate the deflection of the control surfaces and the throttle percentage. On board sensors, such as a pitot tube, barometer, inertial measurement unit, and global positioning system (GPS), continually measure the attitude of the aircraft. These sensor measurements are fused using an Extended Kalman filter in the state estimator to provide measurements for each of the other nodes. ROSflight is used for interfacing with the servos and sensors.

The autopilot and gains were tested in hardware by giving the autopilot various straight-line and orbit paths. The behavior was observed and gains tuned so that the UAS was responsive and did not exhibit instability. Plots of the UAS response to a commanded altitude and course heading during a hardware test flight are shown in Figure 8. The altitude is able to smoothly reach the desired altitude. The course heading is less smooth due to sensor noise and disturbances like wind, but overall it is able to

maintain the desired course.

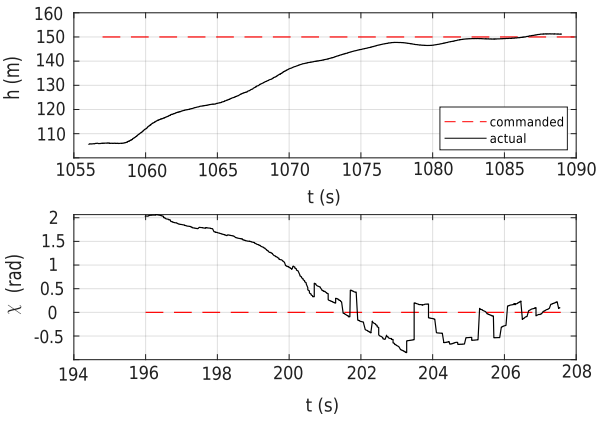

Figure 8: Plots showing the desired altitude (top) and course (bottom) in dashed red and the UAS response in black.

2.3 Obstacle Avoidance

The combined path planning and obstacle avoidance system was designed around the competition specifications. The path planner uses the competition boundaries, obstacles, waypoints, drop location, and search area from the judges' server. When a mission objective is assigned by the ground station, the waypoints needed to accomplish that mission are passed to a rapidly-exploring random tree (RRT) algorithm which plans a path by randomly expanding possible paths. The algorithm is bounded by constraints that ensure the final path is flyable by our UAS, does not hit obstacles, and remains in the competition boundaries. An example of a planned path is seen in Figure 9. The final path is defined by a new set of waypoints that are sent to the autopilot for execution. An overview of our RRT algorithm is given in Algorithm 1.

The path planning system has been tested in both simulation and hardware. For simulation testing, data were gathered over a series of five runs. In each run, ten obstacles and five waypoints were randomly generated within the competition boundaries. A series of waypoints was generated according to Algorithm 1 and given to the autopilot to fly. The results of these tests are summarized in Table 4. On average, the autopilot was able to fly within 3.2 m of the waypoints. There is a single outlier where

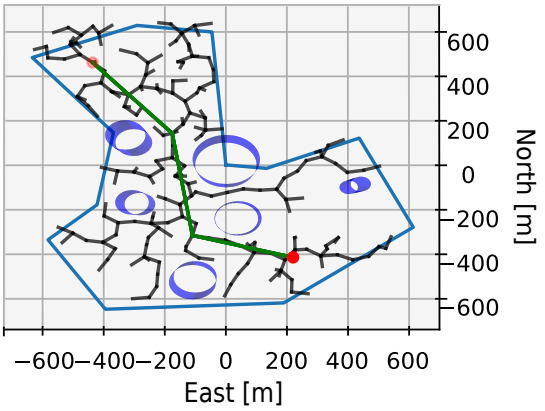

Figure 9: Path Solved by RRT. Green line shows the planned path, black lines show explored paths, light blue lines show competition boundaries, red dots are the beginning and ending waypoints, and blue circles are the obstacles.

Table 4: Minimum distance to five different waypoints as demonstrated in simulation. Results are averaged over five tests. Note that the average error on point four is significantly larger then the other points due to a single outlier.

Simulation Performance	
Point 1	0.60 m
Point 2	0.26 m
Point 3	0.18 m
Point 3	0.43 m
Point 4	14.34 m

the autopilot failed to reach a waypoint and only got within 70 m of it. Removing this outlier gives an average minimum distance of 0.37 m. In all the simulations, only 1 of the 50 total obstacles was hit.

Several hardware tests were performed to validate these simulations results. In these tests, four waypoints were chosen and given to the path planner. The path planner determined a flyable path that connected all four points and passed the path to the autopilot. The autopilot then flew this path. Table 5 summarizes the results of this test.

Overall, we are pleased with the performance of the path planner. It is able to consistently enable the UAS to fly within 1 m of waypoints in simulation and within 10 m in hardware. Although there was an outlier in the simulation testing, our ground sta-

Algorithm 1 Rapidly-Exploring Random Tree

Input: Obastacles O, Boundaries B, start point p_s, end point p_e, node distance D
Output: Waypoint Path W
  successful paths $count = 0$
  Initialize RRT graph as $T = \{p_s\}$
  while $count < 5$ **do**
    $p \leftarrow$ generateRandomNode(B)
    $i \leftarrow$ findClosestNode(B)
    Find segment in direction of P:
    $p_l \leftarrow$ findPath(p, i, D)
    if flyablePath(T, i, p_l) **then**
      T.append(p_l)
    end if
    if flyablePath(T, p_l, p_e) **then**
      mark p_l as a complete path
      $count+ = 1$
    end if
    $newP \leftarrow P$ with length $< SegLength$
    T.append($node_e$)
  end while
  $paths \leftarrow$ smoothCompletePaths(T)
  $W \leftarrow$ findShortestCompletePath($paths$)

Table 5: Minimum distance to four different waypoints as demonstrated in an autonomous hardware test. Results are averaged over four tests.

Hardware Performance	
Point 1	5.95 m
Point 2	6.47 m
Point 3	5.53 m
Point 4	2.68 m

tion is able to replan paths in flight, so we have the ability to reattempt a waypoint if needed.

2.4 Imaging System

The challenge presented for the imaging system is to gather high-quality, stabilized images at our flight altitude. These images must have sufficient resolution to classify ground targets, while being small enough to stream over the network with no data lost or significant delay in transmission. Finally, the camera needs to be carried by the airframe, so tradeoffs between camera weight, size and image quality must be considered.

With these challenges and requirements in mind, the camera chosen is the Sony a6000 with its stock 16-50 mm lens. The a6000 is a consumer grade compact mirrorless camera with a 24 MP sensor and communication over a USB 2.0 interface. At an altitude of 150 ft, this sensor provides three pixels per inch resolution of ground targets. A custom driver with a ROS interface was built to allow image streaming and setting manipulation over WiFi during flight.

Given the advanced autofocus, stabilization, and high shutter speed of the camera, it was determined there was no need for a gimbal within the UAS. The a6000 will point directly out of the bottom of the airframe and stream images to the ground while the plane flies a search path. All autonomous and manual detection, classification and localization processes will happen on the ground station machines.

2.5 Object Detection, Classification, Localization (ODCL)

To address the ODCL problem, a server-client architecture was created. This architecture allows any number of autonomous and manual vision clients while also being easy to set up and maintain. The server-client model addresses concerns with last years system which spread manual detection and classification across multiple machines — making it hard to identify bugs, replicate results and set up quickly. This year's implementation is shown in Figure 10. Data flow between client and server, with the server holding a definitive-final copy of a target image, as well as a history of its state during intermediate detection, classification and submission steps.

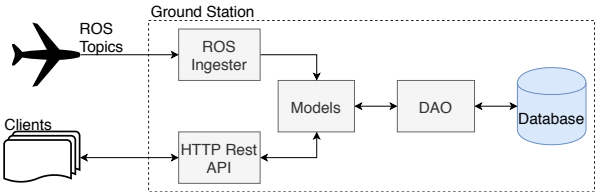

Figure 10: Imaging Server Architecture

2.5.1 Manual Vision System

One of the clients shown in Figure 10 is a graphical user interface (GUI) used to manually crop, classify and submit ground targets. GUI development prioritized intuitive use and modularity. The server-

client architecture allows as many team members as needed to launch the GUI and simultaneously crop and classify images. The GUI contains numerous visual indicators on submission status in order to clarify progress. The following is a top-level description of how the GUI functions.

The cropping tab allows users to pull a raw image from the server, crop a region of that image, and then submit the cropped image back to the server.

The classification tab allows users to pull a cropped image from the server; classify the shape, background color, alphanumeric, alphanumeric color, orientation, and target type; and submit those classification details back to the server.

The server automatically groups multiple images of the same target according to classified type, shape, and alphanumeric. The manual submission tab allows users to pull down each group of classified images, decide which image to submit and optionally override classification information, before submitting the finalized classification back to the server and judges.

2.5.2 Autonomous Vision System

The autonomous vision system (AVS) is divided into two subsystems: region of interest (ROI) detection and object classification. Both subsystems rely on algorithms implemented in OpenCV3, an open-source computer-vision library. The classification system also uses machine learning models to identify letters and shapes. An overview of the entire AVS is shown in Figure 11.

ROI Detection

Given an input image streamed to the ground station from the UAS, the autonomous detection system is able to identify and crop potential targets from the image. First, the image is passed through a meanshift filter. This filter was chosen because it is capable of reducing noise while enhancing color edges in an image, making targets more easily identifiable. The edges of the image are then extracted using a low-sensitivity Canny edge detection algorithm; after which enclosed edges are filled using a flood fill algorithm. The centers of these contours are then found using a simple blob detector. The keypoints returned by the blob detector are cropped out of the original image and passed to the autonomous classification system to be classified or rejected.

Object Classification

Given a cropped image of a potential target, the autonomous classification system either accepts the image as a target and classifies it or rejects the image as a false positive. First, the classification system uses a combination of meanshift filtering, edge detection, and flood filling to find the main contour of the potential target. This contour is passed into a pre-trained SqueezeNet classifier implemented with PyTorch to identify its shape.

Then, using the contour as a mask, the target is extracted from the background of the image and k-means clustering is done to find the two most prominent colors of the image. The larger color is assumed to be the target color and the smaller color is assumed to be the letter color. These colors are then identified by converting them to the L*a*b* colorspace and finding the minimum Euclidean distance from each color classification in a bank of predetermined colors.

After the letter color is known, the letter is extracted from the rest of the image and passed through another pre-trained SqueezeNet classifier for alphanumeric identification. To find the orientation of the letter, matching keypoints are identified between the extracted letter and a template letter image and an affine transformation matrix is created. The angle of the letter is found from this matrix. After all characteristics have been determined, the classification and the cropped image are returned to the server and sent to the judges.

All classifiers contain a no-target classification. If at any point a detected target is classified as no-target, its considered a false-positive and rejected.

2.5.3 Target Geolocation

Both the autonomous and manual systems use the same algorithm to determine the GPS coordinates of targets detected. The script takes in the pixel coordinates of a detected target in the image frame and uses the aircraft's current state to estimate the

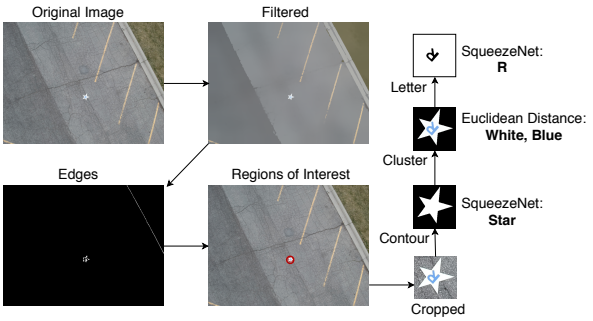

Figure 11: Autonomous Vision System

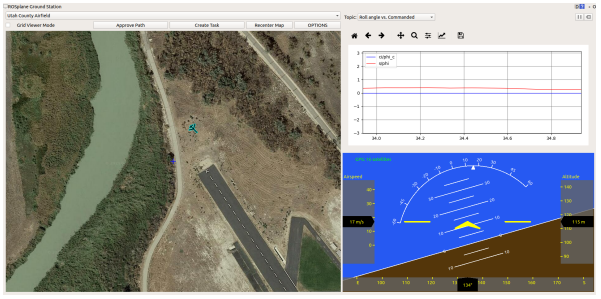

Figure 12: Aircraft ground station GUI during a flight test.

location of the target. The primary challenge for geolocation was properly fusing GPS, image and state (roll, pitch, yaw) information. The final geolocation algorithms were based on those described in "Small Unmanned Aircraft" by Beard and McLain.

2.6 Communications

2.6.1 Communication Protocols

As can be seen from Figure 2, both radio and WiFi will be used to facilitate connection between the subsystems on the ground and in the air. The Ubiquiti M-Series Bullet data link allows for communication between the ground and the aircraft over a 5.0 GHz WiFi network. A 915 MHz radio link between the radio transmitter and receiver allows for manual control and arming/disarming of the aircraft.

ROS and HTTP facilitate the majority of inter-component communication over the WiFi network. ROS, as a Linux middle-ware, allows for real-time communication between machines running individual nodes, or executables, over a WiFi network. In our system, all subsystems communicating via ROS either are ROS nodes to be run on a machine with Linux installed. For more information about ROS nodes and how they communicate over a network, see http://www.ros.org/.

2.6.2 Autopilot Ground Station Interface

The ground station for the autopilot, shown in Figure 12, communicates with the UAS during flight over the ROS network using the Ubiquiti WiFi interface. The ground station is used to receive and send telemetry and waypoint data to and from the UAS, where the data are formulated as ROS messages and ROS service calls.

2.6.3 Unmanned Ground Vehicle Interface

The UGV operates its own telemetry and control link using a 433 MHz, 0.1 W frequency hopping spread spectrum radio. 433 MHz was chosen for its long range at low power, as well as because it will not interfere with our other RF communications. This link supports communication using the MAVLink protocol at 38400 baud.

2.7 Air Drop and Unmanned Ground Vehicle

Four primary challenges were addressed in the air drop and unmanned ground vehicle subsystems: designing payload storage and release mechanisms, achieving a gentle drop, planning the payload release location, and developing the unmanned ground vehicle (UGV).

2.7.1 Payload Storage and Release Mechanisms

The payload storage and release mechanism was designed to increase repeatability and consistency between tests. A linear actuator was used that latches the bay door and opens it on a PWM command. At deployment, the door is free to open from the weight of the UGV. The bay door was designed

with a front hinge and torsional spring to close the door quickly with the additional force of the wind. To prevent the bay door from unexpectedly opening during flight, an arming command must first be sent. To ensure that the release mechanism opens the bay door and the payload falls out when commanded, the UAS was set up in the lab and the mechanism was repeatedly commanded to release. No failures to deploy were seen in the drop test, and in six flight payload drops, only one failure to deploy occurred. After that flight, the cause of this single failure was determined and resolved.

Figure 13: Side view of the parachute and UGV.

2.7.2 Achieving a Gentle Drop

To estimate the proper diameter of our parachute, we used Equation 1, where D is diameter in meters, with parachute coefficient of drag $C_D = 1.5$ (provided by the parachute manufacturer), mass $m = 0.5$ kg (from Section 1.2.4), air-mass density $\rho = 1.22$ kg/m^3, and terminal velocity $v = 3.0$ m/s. A 36-inch parachute was purchased based off these calculations.

$$D = \sqrt{\frac{8mg}{\rho C_D \pi v^2}} = \sqrt{\frac{8 \cdot 0.5 \cdot 9.8}{1.22 \cdot 1.5 \cdot \pi \cdot 3.0^2}} \quad (1)$$
$$= .87\text{m} \approx 34\text{in}$$

Two key risks in using a parachute for the payload drop are if the parachute becomes tangled, it won't open, and if the parachute gets caught inside the airframe, the payload may only partially deploy, inhibiting the remainder of competition flight. Additionally, to be able to predict the landing location of the parachute, it needs to open in a consistent manner. Considering these factors, we developed and recorded a method for folding the parachute that prevents the shrouds of the parachute from becoming tangled during deployment. We also developed a plastic tube that the parachute deploys through, which ensures that the parachute does not become snagged on anything inside the plane. Over three flight tests, we used recorded videos to determine that the time between the opening of the bay door until the parachute was fully open averaged 1.61 seconds, with a standard deviation of 0.16 seconds. These tests show that our parachute deploys reliably and consistently.

2.7.3 Payload Release Location Planning

The calculations to determine the payload release location have been prioritized to be simple and consistent. The payload release location is calculated considering two distinct regions determined by experimentally dropping a GPS tracker: closed parachute and open parachute descent.

The closed parachute decent region assumes that the payload maintains the velocity of the plane for a constant amount of time which is determined through experimentation, as described in Section 2.7.2.

The open parachute decent region assumes that the payload drops at a constant rate and it moves with the estimated north and east wind velocity. After the payload dynamics of these two regions are calculated, they are combined to return a specific drop location. A graph of the calculated payload movement in these regions is plotted in red in Figure 14.

The electronics on the UGV include a 500 mAh lithium ion battery; a 5 V regulator; a microcontroller board with integrated inertial sensors, an altitude sensor, and an integrated 433 MHz radio; and a GPS receiver. The 433 MHz radio is used to transfer telemetry from the UGV and to terminate the UGV from the ground station. The position estimation on the UGV is done by fusing data from the inertial measurement unit and GPS, and is fed into a basic PID controller that attempts to minimize the difference between the UGV and the target drive location. The total mass of the UGV

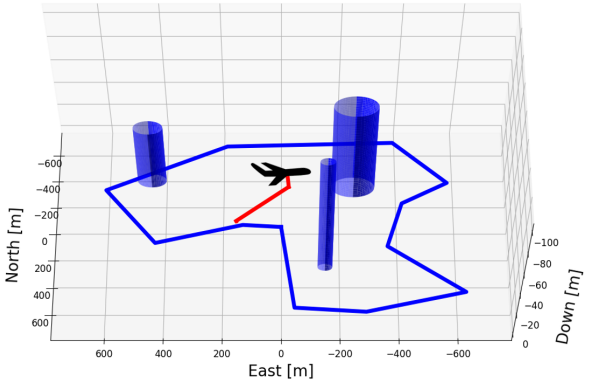

Figure 14: Payload release planning with boundaries and obstacles in blue and payload movement in red.

is measured to be 0.42 kg, well within competition specifications, and near the 0.5 kg value used to calculate the optimal parachute diameter. Testing has shown that the fully loaded UGV has a maximum speed of 7.5 mph, within the competition max speed of 10 mph.

2.8 Cyber Security

Due to the reliance of the UAS on WiFi, radio, and open-source software, the system is vulnerable to cyber attacks on multiple fronts. Table 6 details principle access points and vulnerabilities in the UAS, as well as strategies for mitigation in the case of cyber attacks. The failsafes described in Section 3 are an important part of our mitigation strategy.

3 Safety, Risks, and Mitigations

As a team, we recognize the critical importance of safety, including compliance with the Academy of Model Aeronautics (AMA) safety code and competition rules. To mitigate any safety risks or mission failure risks, our team performed an in-depth failure modes and effects analysis (FMEA). For this analysis, an extensive list of our system's failure modes was created via a brainstorming session. Failure modes are defined as high-level events in which a system or subsystem fails to perform its intended function and/or jeopardizes human safety. Over 70 different failure modes were identified for the entire system. Following the creation of this extensive list, the potential causes and effects of each failure mode were listed. The severity, likelihood, and detectability of each potential failure, cause, and effect combination were then rated on a scale of 1 to 10, with 1 being the least severe, least likely, and most detectable, and 10 being the most severe, most likely, and least detectable. These three ratings were multiplied to obtain a risk priority number (RPN) for each potential failure, cause, and effect combination. A severity rating over 8 or RPN over 125 was deemed unacceptable, and a method to mitigate that risk was developed. The following sections detail our efforts to mitigate risks during both the development process and during a mission demonstration. Upon performing these mitigation efforts we reanalyzed each potential failure, cause, and effect combination to ensure it had been corrected to within acceptable levels.

3.1 Developmental Risks and Mitigations

During development we have implemented strict safety standards which have kept us and others safe. Our safety pilot has over 20 hours of combined simulated and real remote control flight experience and is a Federal Aviation Administration certified drone pilot. Care is taken to avoid restricted airspace, avoid flight over people, avoid flight over property that could be damaged, and avoid violation of the AMA safety code. While safety is everyone's responsibility, one team member has been specifically assigned as our team's safety officer. This team member has led our team in tasks such as battery safety education. As can be seen in our developmental FMEA (Table 7), any major developmental risks have been mitigated.

3.2 Mission Risks and Mitigations

Table 8 shows the mission FMEA for several of the pertinent misssion failure modes. As can be seen, the most common method we used to mitigate risk was the field flight checklist (FFCL). This checklist allows us to rigorously check the aircraft for airworthiness prior to flight. After implementing this FFCL the percent of test flights ending in a crash fell drastically from 100% to less then 5%.

Through these analyses, we have significantly decreased the likelihood of a system failure causing

Table 6: Cyber security risks, impacts, prevention, and mitigation for the UAS.

Vulnerability	Risk	Impact	Mitigation
5.0 GHz WiFi Data Link	Unauthorized access to the network	Reduced bandwidth, network disabled, data compromised	WPA2 encryption on WiFi, required login
915 MHz RC Link	Interference, jamming	Safety pilot loses contact with UAS, unauthorized pilot takes control of UAS	On RC loss the autopilot automatically loiters in place, eventually descending to the ground. RC receiver and transmitter are paired, preventing unauthorized access.
433 MHz UGV Link	Interference, jamming, unauthorized access	UGV is stopped prematurely	Password stored on ground station is used to stop UGV
GPS Module	Spoofing, jamming	Loss of location estimate, degraded attitude estimate	GPS receiver includes advanced spoofing and jamming prevention. State estimation reduces trust in GPS information.

property damage, personal injury, or mission failure. As such we are confident that we will be able to complete a mission demonstration successfully on competition day.

4 Conclusion

We are confident that the system presented above will not only be able to perform the assigned tasks, but do so admirably. We have have designed our system to meet the specific mission requirements, we have proven the effectiveness of our design by testing its hardware and software extensively, and, as much as is reasonable, we have mitigated risk. Figure 15 shows the UAS in flight during performance testing. We look forward to demonstrating the performance of the UAS at the competition this summer.

Figure 15: Images of the UAS in flight.

Table 7: Abbreviated developmental FMEA. As shown, all pertinent safety risks have been mitigated.

Component	Functional Purpose	Failure Mode	Failure Effect	Failure Cause	Previous Situation				Assigned Action	Improved Situation			
					S	L	D	RPN		S	L	D	RPN
Battery	Provide current to all systems in the air	Battery Fire	Damage to property and risk to human life	Charging or storage error	10	2	3	60	Assign safety officer	10	1	2	20
Flight Boundaries	Limit flight to safe areas	Dangerous crash	Damage to property and risk to human life	Flight boundary violation	10	3	2	60	Assign safety pilot	10	2	1	20
				Unsafe flight boundaries	10	2	1	20	Assign safety pilot	10	1	1	10
Personal Safety Equipment	Inhibit personal injury	Foreign object in eye	Loss of sight	Foreign objects near props	10	4	3	120	Establish protocol to keep props clear	10	2	2	40
				Safety glasses not applied	10	7	3	210	Add safety glasses to field flight checklist	10	3	3	90
Propulsion System	Provides thrust to aircraft	Motors start while hands are near props	Abrasions and lacerations	Lack of communication	10	6	5	300	Establish protocol to yell "clear props" whenever starting motors	10	1	1	10
				Arm disarm switch failure	10	4	6	240	Require system to be powered off or props removed to work near motors	10	1	6	60
Takeoff & Landing Procedure	Facilitate beginning and end of mission	Aircraft impacts person or property	Damage to property and risk to human life	Unsafe aircraft flight path	10	3	3	90	Require all personnel to be behind aircraft flight path before takeoff and landing	10	1	3	30

S: Severity of failure effect
L: Likelihood of failure occurring
D: Detectability of cause before failure occurs
RPN: Risk Priority number (S*L*D)

Table 8: Abbreviated mission FMEA. Several potential failure, cause, and effect combinations were found to have greater than acceptable risk priority numbers (RPN) and/or severity ratings. As such, the actions shown in the Assigned Action column were used to mitigate this risk. Improved ratings are included under "Improved Situation".

Component	Functional Purpose	Failure Mode	Failure Effect	Failure Cause	Previous Situation				Assigned Action	Improved Situation			
					S	L	D	RPN		S	L	D	RPN
RC Transmitter	Communicate Commands from the RC Pilot to the RC Receiver	Hardware Failure*	Aircraft Loiters	Poorly connected circuit	8	2	7	112		8	2	3	48
		Transmits incorrect data	Crash	Settings incorrect	9	2	6	108	FFCL**	9	1	4	36
		Loss of Connection	Aircraft Loiters	Settings incorrect	8	6	8	384	FFCL	8	4	3	96
				Interference	8	4	9	288	FFCL	8	4	3	96
				Transmitter battery dead	8	6	3	144	FFCL	8	4	2	64
Airspeed Sensor	Measure Va	Software Failure	Flight Less Smooth	Internal code	4	1	10	40		4	1	10	40
		Inaccurate Readings	Flight Less Smooth	Plugged pitot tube	4	4	5	80		4	4	5	80
				High angle of attack	4	4	2	32		4	4	2	32
				Incorrect mounting	4	2	2	16		4	2	2	16
		Hardware Failure	Flight Less Smooth	Poorly connected circuit	4	1	7	28		4	1	7	28
GPS	Measure global position	Software Failure	Crash	Internal code	9	3	10	270	Extensive testing**	9	3	3	81
		Inaccurate Readings	Crash	Interference	9	4	5	180	FFCL	9	2	4	72
		Hardware Failure	Manual Landing	Poorly connected circuit	6	1	7	42		6	1	7	42
Battery	Provide current to all systems in the air	Loss of Power	Crash	Battery not charged correctly	9	5	3	135	FFCL	9	5	2	90
				Battery degradation	9	1	1	9	FFCL	9	1	1	9
Motors	Rotate Props	Overheat	Fire and Crash	Over stressing the motors	10	3	5	150	Add warning to FFCL	10	2	5	100
		Does Not Transmit Torque	Glide to Landing	Props unsecured	7	8	3	168	FFCL	7	5	2	70
		Rotates the Wrong Way	Mission Does Not Start	Wires connected backwards	6	3	2	36	FFCL	6	1	2	12
		Hardware Failure	Glide to Landing	Poorly connected circuit	7	1	7	49		7	1	7	49
Wiring	Transmit power and signals	Does Not Transmit Electricity	Crash	Wires connected to incorrect ports	9	7	8	504	FFCL	9	3	3	81
			Crash	Electrical short circuit	9	3	8	216	Shrink wrap all wires	9	1	8	72
			Crash	Electrical open circuit	9	8	5	360	FFCL	9	8	1	72
Airframe Body	Contain components, provide lift, provide stability, & respond to control inputs	Flight Characteristics Change	Crash	Icing	9	2	1	18	Only fly in good weather	9	1	1	9
				Components move	9	5	5	225	Strap down components	9	3	3	81
		Parts Break Off	Crash	Flight envelope exceeded	9	2	3	54	Train safety pilot	9	2	2	36
				Poor manufacturing	9	6	7	378	Extensive testing	9	6	2	108
				Part poorly attached	9	2	7	126	FFCL	9	2	3	54
				Unidentified flying object impact	9	1	3	27	Train safety pilot	9	1	3	27
WiFi Light Beam	Transmit data between WiFi router and the WiFi receiver	Loss of Connection	Manual Landing	Interference	6	8	7	336	FFCL	6	3	6	108
				Antenna not pointed correctly	6	10	3	180	Assign antenna pointer	6	3	3	54
		Hardware Failure	Manual Landing	Poorly connected circuit	6	1	7	42		6	1	7	42
		Software Failure	Manual Landing	Internal code	6	1	10	60		6	1	10	60
Human Operators	Give high level commands & ensure safety of flight	Sick	Mission Does Not Start	Bacteria or viruses	5	4	3	60		5	4	3	60
		Can Not Attend	Mission Does Not Start	Other plans	5	1	1	5		5	1	1	5
		Sends Incorrect Commands	Crash	Poor judgement	9	2	9	162	Extensive practice	9	1	9	81
			Crash	Poor understanding of system	9	2	5	90	Extensive practice	9	1	5	45

* "Hardware Failure" refers only to electrical hardware (e.g. USB port breaks)
** FFCL is the Field Flight Checklist to which we add items to test and do before flight
*** Extensive testing before use refers to extensive flight tests before the competition. We currently perform flight tests a couple times a week.

S: Severity of failure effect
L: Likelihood of failure occurring
D: Detectability of cause before failure occurs
RPN: Risk Priority number (S*L*D)