# Cornell University Unmanned Air Systems
## 2016 AUVSI Student UAS Competition

# Journal Paper


Figure 1: Theseus UAS

**ABSTRACT**
Cornell University Unmanned Air Systems' (CUAir) objective for the AUVSI SUAS 2016 competition was to design a highly modular system able to be manufactured, tested and rebuilt in a parallel fashion. The team's goal was to effectively design and build the system using knowledge from previous years and surpass the flight and software performance of previous year's systems. CUAir's Unmanned Aerial System (UAS), Theseus, was a joint effort of 50 undergraduate students within the fields of Computer Science, Electrical Engineering, Mechanical and Aerospace Engineering, and Business. Using knowledge from all fields, CUAir's members manufactured a custom airframe with modular and easily transposable parts, a robust power system, a fully modular and tested distributed vision system, an autopilot, and a custom ground station. Throughout the year CUAir stressed testing and reliability within the modular framework to ensure that each piece of the system was safe and reliable.

# Contents

# 1 Systems Engineering Approach

## 1.1 Mission Requirement Analysis

CUAir's modular UAS design required extensive planning and task scheduling to ensure completion of all primary and as many secondary tasks as possible while meeting the team's safety and reliability standards. Using the following criteria, the team prioritized task completion and testing:

**Construction:** the amount of time necessary to complete, test, and iterate a project
**Point Value:** the estimated score in competition for completing the task
**Testability:** the degree of reliability and safety to which the system could be tested
**Knowledge:** the experience the team has working with a system for this task

The team decided on:
Essential tasks - tasks the team has completed in the past and are confident will finish testing by competition
Auxiliary tasks - tasks not completed previously by the team but will be developed and tested
Non-essential tasks - tasks deemed plausible, but not immediately worth the risk to the system

**The essential tasks are:** Autonomous Navigation, Takeoff and Landing;Interoperability; Manual Target Detection, Localization, and Classification (MDLC); SRIC; Off-Axis Target (OFAT); Emergent Target; Actionable Intelligence
**The auxiliary tasks are**: Autonomous Target Detection, Localization, and Classification (ADLC); Search, Detect, and Avoid (SDA)
**The non-essential tasks are** Airdrop

## 1.2 Design Rationale

Theseus was designed with CUAir's philosophy of simplicity, modularity, and durability. The team's goal was to design a system that was simple to use on the field, robust to crashes, repairable in the field, and quick to manufacture. From these goals, the following subsystem rationales were conceived:

| Goal | Description | Improvements |
|---|---|---|
| **Improved Software Design** | • Software components are to be independently developed and tested<br>• Components should be independent of the implementation of other components | **Modularity/RESTful API** - Predefined interfaces ensure parts can be developed independently |
| **Improved Software Testing** | • Software must be fully unit and integration testable<br>• All changes are to be tested before being used in production | **Continuous integration** - Ensures software is functional, robust, and compatible |

| | | |
|---|---|---|
| **Improved Hardware Design** | • Mechanical design must allow easy replacement and modifications of airframe components or payloads<br>• Designs must be able to be constructed in parallel and identically | **Modularity**: All components are designed to be easily replaced, modified, and transported<br>**Improved Manufacturing**: All components employ manufacturing techniques that ensure repeatability of designs |
| **Rigorous Hardware Testing** | • Mechanical designs must be tested for strength and functionality<br>• Analyze stresses on the aircraft and arrive at the right selection of materials and manufacturing techniques | **Rapid prototyping** - Iterate through multiple design configurations |
| **Rigorous Electrical Testing** | • Electrical boards must be able to sustain high shock on takeoff and landing, extended current draw and burst current draws, and exposure to extreme weather conditions | **Stress Tests** - Power cycling and forcing high current draw on powerboard under load<br>**Extensive Tests** - Ensured functionality under any foreseeable conditions |
| **Design for Failure** | • Mechanical and electrical components must be designed to reduce severity of failures<br>• Designs to aid mitigation techniques | **Incorporate failure points:** Allow less critical components to take impact on crashes **Modularity:** Modular design that allows for immediate replacement of damaged/faulty components |

Table 1: Design Rationale

## 1.3 Expected Task Performance

Compared to its sister systems of similar aerodynamic characteristics, Theseus has shown to be far more stable in mock missions. The system has been proven successful in both benchtop and flight scenarios for all the primary and secondary tasks we will attempt. Theseus has been tested in dozens of flights, proving its capability of autonomous takeoff, autonomous navigation along the flight path with a waypoint accuracy of 50 feet, flying in a search pattern, and autonomous landing. The autopilot system is capable of recognizing stationary and moving obstacles and avoiding them. The system has achieved interoperability rates of 20Hz. CUAir's custom ground station has been tested for in-flight re-tasking, downloading and displaying server information, time, and obstacles at 10Hz and uploading targets and related details while in flight. The onboard SRIC system is able to automatically detect the ground SRIC server and upload and download information to and from it. CUAir's distributed software systems is able to manually and autonomously detect and provide all 5 features for targets. Additionally, Theseus's gimbal enabled the system to locate off-axis targets. Although Theseus was designed to complete the airdrop task, the team was not permitted the area to safely test the system. Overall, Theseus is expected to complete every primary, and all but one task at objective levels.

## 1.4 Programmatic Risks and Mitigation Methods

During the construction phase, the team performed a part-by-part analysis of the system and testing plan. For each part, we established a plan for mitigation. The following table represents the major risks we found,

the analysis of their impacts and probabilities, and how they could be tempered.

| Risk Factor | Description of Risk | Impact | Likelihood | Mitigation Method |
|---|---|---|---|---|
| **Project completion delays** | Delay in the completion of projects could push other projects past our final testing date. | Medium | High | Reuse existing systems developed in the previous years Build in buffers into the schedule |
| **Integration Issues** | As each system is developed and tested in parallel, integrating each subsystem could reveal issues individual systems did not have. | High | Medium | Each system was developed on with a hard API to other systems Subsystems are chosen based on compatibility |
| **Damage to the Airframe** | Delay in flight testing due to crashes or damage during transportation | High | Low | Maintain spares of airframe and mechanical components Implement a modular design that allows for interchangeability |
| **Damage to the Avionics** | Schedule set-back due to time required to procure new avionics components in addition to the financial overhead | High | Low | Maintain spares of avionics components |

Table 2: Programmatic Risks and Mitigation Methods

# 2 UAS design

## 2.1 Airframe

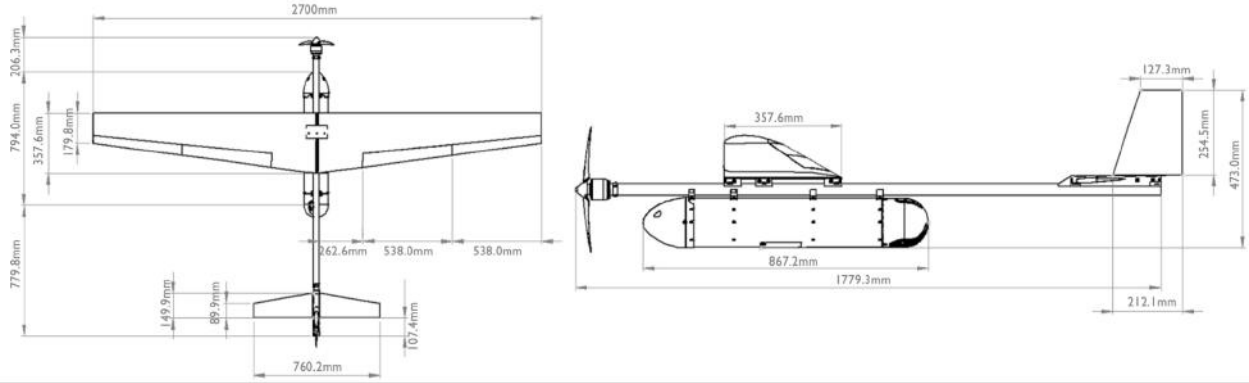


Figure 2: Aircraft Projections, Dimensions Shown

### 2.1.1 Overall Design

Theseus's airframe is based on a 'pod plane' design where each component is easily replaceable or adjustable in the event of a mechanical failure. All components of the airframe are attached to a central boom. This allows for various combinations of installed payloads, and facilitates movement of the center of gravity forward and backward relative to the wings and tail. Additionally, the aircraft is built to withstand high

shock forces associated with catapult launch and controlled belly landing. With a desired static margin of 0.20, mean flight speed of 20 m/s, and total weight of 10.4kg, the team wrote a genetic algorithm to optimize flight characteristics. Theseuss main aerial characteristics are tabulated below.

| | Wing | Vertical Stabilizer | Horizontal Stabilizer | Aircraft Dimensions and Specifications | | Performance | |
|---|---|---|---|---|---|---|---|
| Airfoil | MH-114 | Drela HT-12 | Drela HT-12 | Length(m) | 1.80 | Cruise Speed (m/s) | 10 |
| Span (m) | 2.70 | .255 | .712 | Width (m) | 2.70 | Max Speed (m/s) | 25 |
| Area ($m^2$) | .647 | .0432 | .0861 | Height (m) | 0.47 | Min Speed (m/s) | 15 |
| Avg. Incidence Angle (°) | 2.00 | 0.00 | -2.21 | Weight (kg) | 9.53 | Min Turn radius (m) | 20 |
| Aspect Ratio | 10 | 1.5 | 6 | Prop. Size (in x in) | 17x10 | Flight Hours (hrs) | 7.5 |
| Lift (N) | 102.5 | 0 | -2.93 | Motor | Axi 5320/20 | Flight Time (min) | 30 |
| Mean Aero Chord (m) | .264 | .173 | .122 | Payload Weight (kg) | 3.22 | Assembly Time (min) | 5 |

Table 3: Overall Design Specs

### 2.1.2 Wing

The wings maintain a 2° dihedral and washout from 3° to 1°, as well as a taper ratio of 0.5. The 2° dihedral gives the aircraft a natural tendency to level itself when in the air, and the washout ensures that the root section of the wings stall before the tip. This feature allows the pilot to maintain roll control on an onset of a stall. From comparing airfoil characteristics in Xfoil, the team decided to use the MH-114 airfoil due to its high lift-to-drag ratio and gentle stall characteristics at high angles of attack. The resulting airfoil characteristics were used as the input to the teams custom made genetic algorithm which calculated the desired dimensions and locations of the wing and tail. This algorithm took upper and lower bounds specified by the user (such as tail angle of attack, wing planform area, and the location of the center of gravity), constants characterizing the aircraft (weight, flight speed...etc) and desired pitching moment and static margin. The algorithm then computed the stability and design equations until the desired pitching moment and static margin were satisfied, while still satisfying the input bounds and stability equations. The resulting manufactured wing is depicted in figure 3.


Figure 3: Theseus Wing

### 2.1.3 Tail

The tail is designed with a $-2^\circ$ angle of attack, which creates an upward moment about the aircrafts center of mass when flying. This negative angle of attack helps ensure that during a stall, Theseus will naturally pitch down and regain speed. In addition, the vertical stabilizer is offset by 108mm behind the horizontal stabilizer allowing air to flow undisturbed from the horizontal stabilizer over the vertical control surface, even in an event of a planar stall. This undisturbed airflow allows the rudder to maintain the yaw of the aircraft, which when paired with Theseus's propensity to pitch down during a stall, allows for a very easy and safe stall recovery. This not only prevents Theseus from a stall based crash, but more importantly ensures the payload safety. The tail was also designed to break elegantly in the event of a crash such that the control surfaces are not harmed. In order to accomplish this task, all three surfaces are mounted to a tailbox via wooden inserts and one 3D printed break-point, which in turn is fastened with hose clamps to the boom. In the event of a crash, this breakpoint will snap, freeing the surfaces so they do not shear or compres. Both the tailbox and breakpoint can be seen in figure 4. This design allows the entire tail to be replaced in under 10 minutes without affecting other components of Theseus and allowing continuous integration and testing of the mechanical systems.

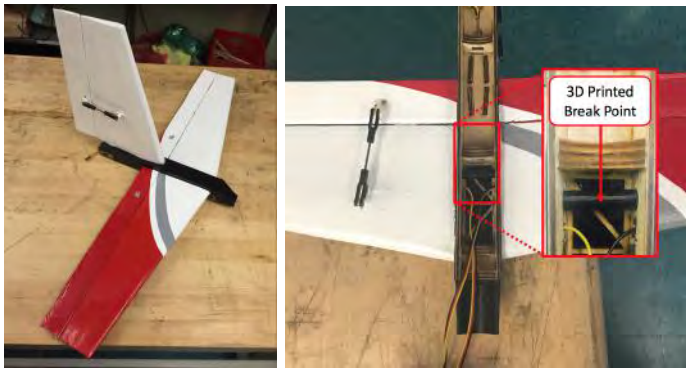


Figure 4: Left:Isometric View of Tail; Right: 3D-printed Break Point

## 2.2 Fuselage

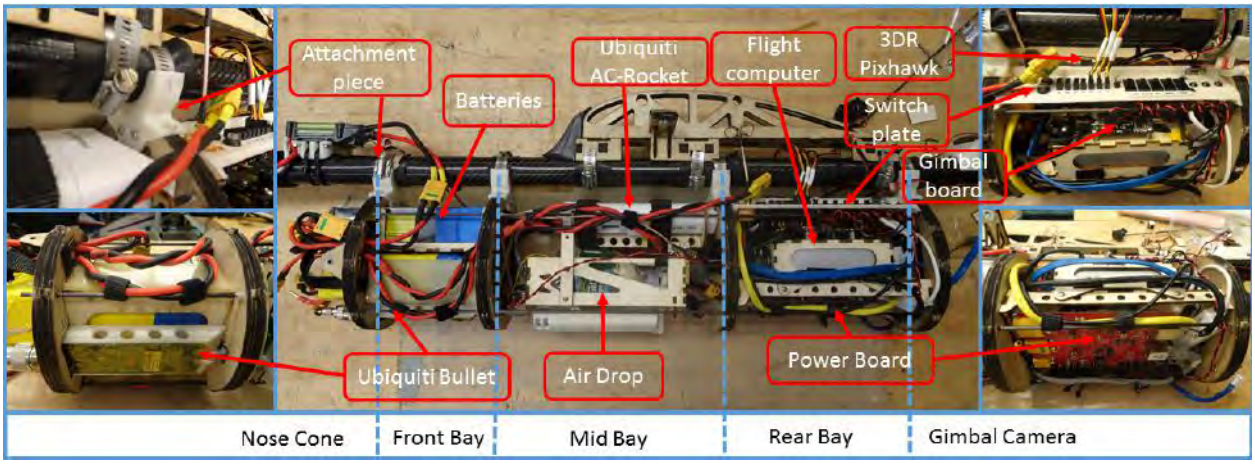


Figure 5: Layout of payloads in the fuselage

The pod-based fuselage was designed to be highly modular for easy assembly and upgradability. The design takes advantage of rapid prototyping techniques such as 3D printing and laser cutting to construct a fuselage in only one week. Four wooden ribs serve as the structural supports and divide the fuselage into three compartments: front bay - batteries and Ubiquiti Bullet, mid bay - Ubiquiti AC-Rocket and airdrop, and rear bay - flight control systems(Pixhawk), power distribution board and flight computer. In addition to the three compartments, an aerodynamic nose cone is installed in the front to protect the batteries, and a rear dome is attached at the back to house the camera and gimbal. The fuselage is attached to the central boom by four 3D printed attachment pieces. A hose clamp goes through each attachment piece and around the central boom securing the fuselage assembly to the central boom. A piece of 0.5mm thick shock-proof polycarbonate wraps around the ribs to act as the fuselage skin and is secured to the attachment pieces with cotter pins and through pegs on the ribs.
The fuselage was designed completely in SolidWorks Computer Aided Design software and was produced using 3D printers and a laser cutter. The nosecone and dome are 3D printed with ABS plastic and utilize a mesh structure to be both light and strong. Additionally, the nose cone attaches to the fuselage via captive panel thumbscrews which allow it to be removed easily without tools. The ribs were made by gluing four circular pieces of 3.1mm birch plywood together. These ribs are then connected by four 3mm diameter carbon fiber spars which run through the corners of all four ribs to sustain the torsional and bending stresses on the fuselage. Other payload bay components were made out of 3.1mm poplar plywood and connect to the rib superstructure via a combination of screws and glue. The clamp attachment method for the fuselage allows its position along the central boom to be adjusted to achieve the desired center of gravity.

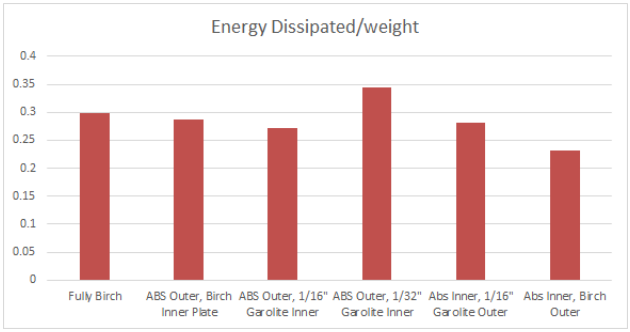


Figure 6: Results from energy dissipation tests

## 2.3 Payloads

The payload system is comprised of an Intel NUC onboard camera (OBC), two communication modules, Ubiquiti Rocket AC and Ubiquity Bullet M2-HP, a SimpleBGC gimbal unit, a Point Grey Flea 3 machine vision camera, and the 3DR Pixhawk Autopilot System.

### 2.3.1 Electrical Diagram

Electrical system diagram below in Figure 7.

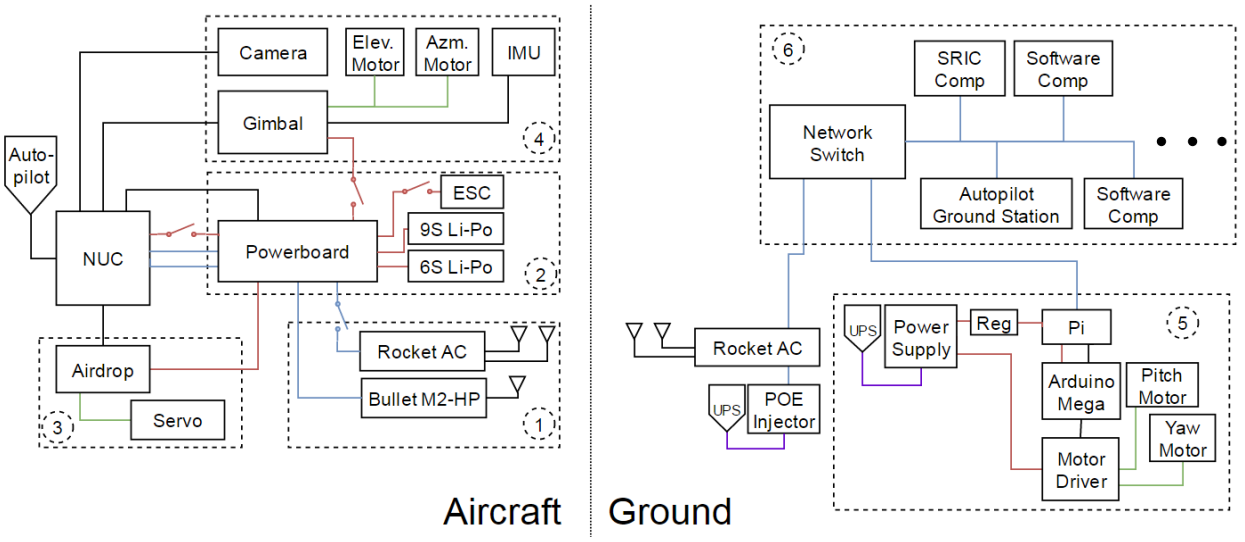


Figure 7: Electrical system diagram of the aircraft and the ground

Red wires signify power connections, black wires signify data or logic, blue wires represent ethernet, green wires represent a motor or servo connection, and purple wires represent wall AC. The airdrop board's servo is powered off of 5V, the gimbal motors off of 12V, the NUC off 12V, and the Rocket AC and Bullet M2-HP are powered through POE via 24 volts. Switches are noted in the connections. Refer to the Autopilott System section for Autopilot Diagram

### 2.3.2 Onboard Computer

The Intel NUC houses a multicore Intel Core i5 processor clocked at 1.9 GHz. This provides the power to run multiple servers onboard Theseus to control the camera, gimbal, power board, and the communications links.

### 2.3.3 Gimbal and Camera

The Point Grey Flea3 machine vision camera with the Tamron M118FM08 lens has a maximum resolution of 4096x2160. Given an average flight height of 250ft, the camera can provide a resolution of 0.53 inches per pixel, which is sufficient to resolve the smallest target features. Given an average flight speed of 20 m/s, the camera needs to take images at a rate of 0.5 images/s to provide complete ground coverage. However, the camera can take up to 4 images/s, and the frame rate is set to 1 image/s during flight.

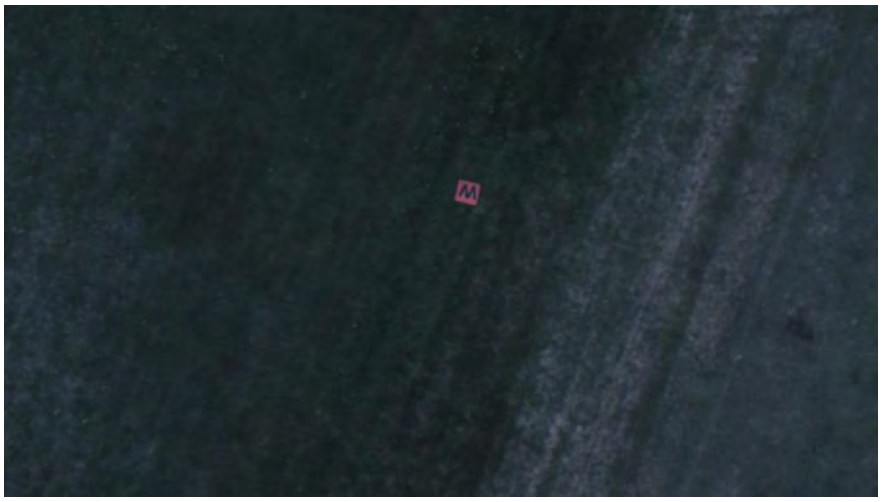

Figure 8: Camera image of target taken during test flight

The gimbal is a two-axis camera stabilization system used to remotely control and point the on-board machine vision camera. Electrically, the gimbal system consists of two brushless gimbal motors controlled by the BaseCam Electronics SimbleBGC 32-Bit board which stabilizes the motor based on two IMU's. Mechanically, the system consists of a 3D printed two-axis system consisting of a roll cage, a pitch arm, and an outer dome to protect the entire system. A gimbal server is run on the on-board computer which facilitates its control. The software controller is configured to use four separate gimbal states: retracted, point at ground, point at angle, and point at GPS. The retract state was designed to point the camera at an angle such that during launches and landings, the camera would be protected. In point at ground state, the gimbal simply points at ground while stabilized. In point at angle state, the gimbal points at a given quaternion. Finally in point at GPS state, the gimbal points at a given GPS position on the ground.

#### 2.3.4 Powerboard

The power board regulates the voltage from a 6S Lithium-Polymer (LiPo) battery and distributes it to the various electrical payloads of the aircraft. When Theseus is on the ground, the power board allows for hot-swap of power from the internal battery to a ground station power supply, without interrupting the functionality of the system. Built in circuitry allows for charging of the 6S battery from the ground station power supply. The board also breaks the connection between the 9S LiPo motor battery and the ESC as a safety feature for when the aircraft is on the ground. Additionally, the power board measures the current being supplied to each system and reports it for logging on the payload computer. A switch plate built into the wall of the fuselage of the aircraft can independently cut power to each of the payload system for safety and compliance with the rules.

## 2.4 Autopilot System

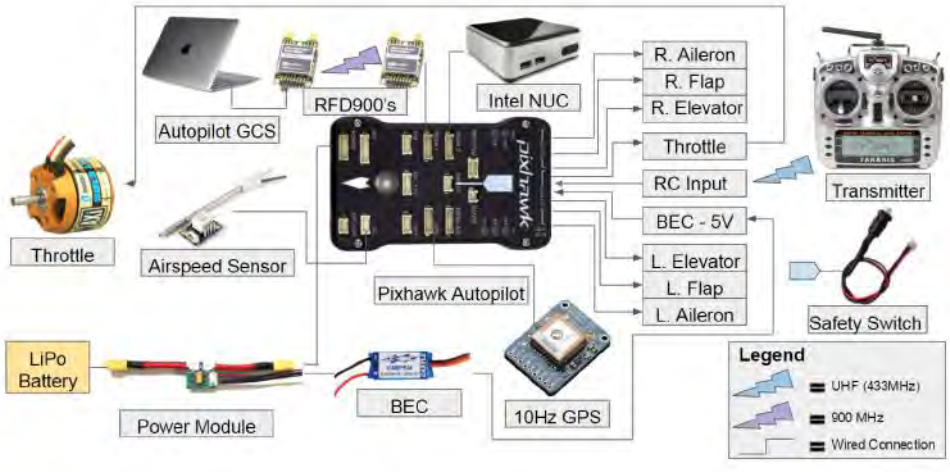


Figure 9: Autopilot system diagram

### 2.4.1 Air System

The 3DR Pixhawk module was selected as the on-board autopilot system because it is the leading open-source autopilot system  a feature that the team deemed as important due to interoperability - and has a highly extensible module system that can easily integrate with a variety of sensors. The open-source codebase allowed us to program the correct flight termination for the competition: If primary RC link is lost, Theseus will abandon its mission to circle in place for 30 seconds waiting for a restored link. If after 30 seconds it still has no link the autopilot will begin a Return to Launch. Finally, if after 3 minutes there is still no link, Theseus will execute a controlled aerodynamic termination. Theseus is capable of fully autonomous takeoff from the teams pneumatic catapult. Utilizing customized ArduPlane firmware, Theseus waits until there is an acceleration of at least 1G, then starts throttle. Theseus then maintains heading and pitches up to 10 degrees until the required takeoff altitude is achieved. Theseus is also capable of fully autonomous belly landing, without the use of expensive laser altimeters or unreliable sonar sensors. The autopilot team was able to work with the airframe team to determine the best landing parameters to ensure that Theseus flares 0.5 seconds before landing and changes its decent speed from 8m/s to 0.25m/s to ensure a soft and reliable landing.

### 2.4.2 Ground Control System

The ground station uses the MAVProxy communication system to interface with the Pixhawk. To overcome efficiency issues that plagued previous web-based implementations, the ground station was switched to the React/Flux architecture with long-polling to make the frontend react in much closer to real-time. The MAVProxy library permits highly modular extensibility and makes it easier to add functionality, including SDA and the web server for the Graphical Frontend through the integration of the module system. Lastly, robustness was confirmed using a mix of back-end tests over a custom designed REST API and automated frontend selenium tests  testing both our API's and graphical system. The primary motivation for building a new web-based ground station rather than using existing ground stations was the cross-platform nature of web-based implementations as well as the need to display and use interop information in the ground station, such as obstacles for SDA. Further, having a modular custom-built ground station makes future extensibility and feature addition significantly easier than trying to adapt an existing system to the competition's

requirements.

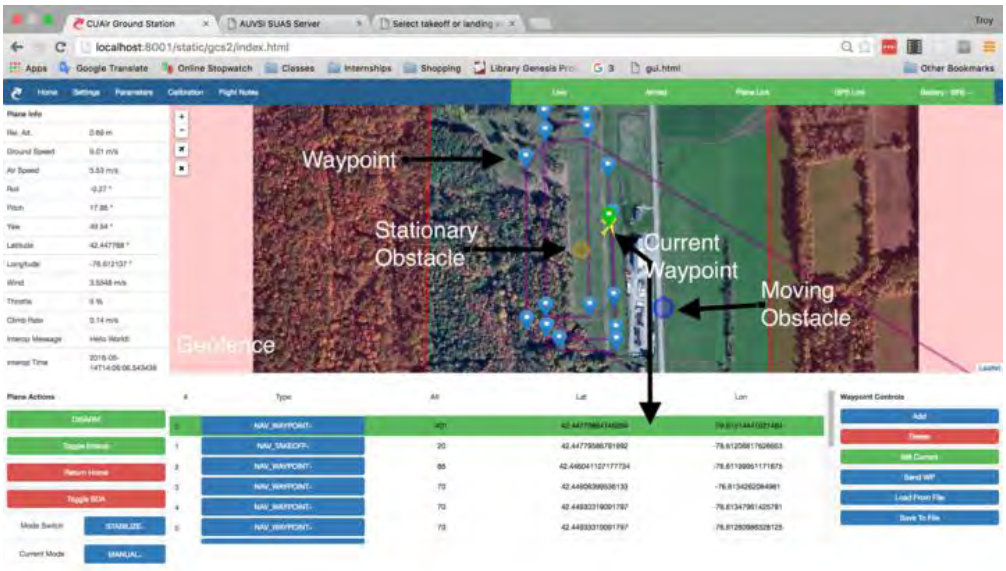


Figure 10: Autopilot Ground Stations Screenshot

### 2.4.3 Sense, Detect, and Avoid

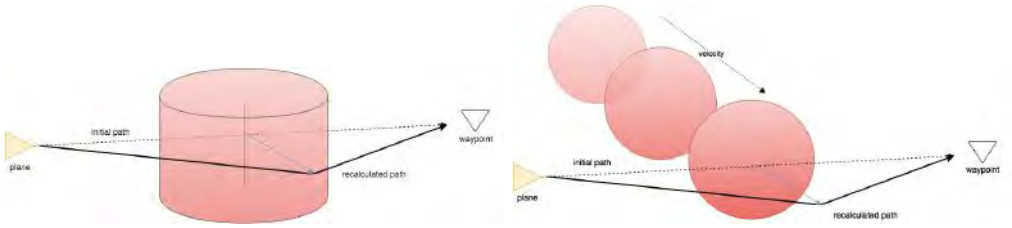


Figure 11: Sense, detect, and avoid Left: Stationary obstacle, Right: Moving obstacle

To complete this task, the team developed a reactive algorithm to anticipate flight path and predict obstacle locations. The algorithm creates a 3D geometric model with flight paths and moving obstacles as point entities on linear trajectories to the next waypoint in the flight plan and stationary obstacles as lines with lengths equal to their height. Our algorithm identified potential obstacle collisions by calculating minimum distance between the linear entities using the linear closest point of approach (CPA) for moving obstacles and the closest point of two 3D line segments for stationary obstacles. CPA assumes constant velocity vector $\mathbf{u}$ for the obstacle and constant velocity vector $\mathbf{v}$ for the plane and is defined as $d(t_{CPA}) = |\mathbf{w}_0 + t_{CPA}(\mathbf{u} - \mathbf{v})|$ where $\mathbf{w}_0$ is the distance vector between the initial positions of the plane and obstacle. Time of CPA, $t_{CPA}$, is calculated as follows.

$$t_{CPA} = \frac{-\mathbf{w}_0 \cdot (\mathbf{u} - \mathbf{v})}{|\mathbf{u} - \mathbf{v}|^2} \tag{1}$$

To detect collisions with stationary obstacles, the ground station models the plane's flight trajectory to the designated waypoint as a line segment $\mathbf{r}(t) = P_0 + t\mathbf{v}$ and the center of the stationary obstacle as a line

segment $\mathbf{q}(s) = Q_0 + s\mathbf{u}$ where $P_0$ is the initial position of the plane, $s$ and $t$ are length variables, $\mathbf{u}$ and $\mathbf{v}$ are direction vectors and $Q_0$ is the zero altitude location of the obstacle. The distance equation is derived as follows.

$$d(\mathbf{r}, \mathbf{q}) = \frac{|(\mathbf{Q_0} - \mathbf{P_0}) \cdot (\mathbf{u} \times \mathbf{v})|}{|\mathbf{u} \times \mathbf{v}|} \tag{2}$$

Distances less than the obstacle's radius for either equation are considered collisions.

Once collisions are detected, a line (A) between the flight trajectory and the projection of that line onto the center of the obstacle is calculated. The algorithm iteratively calculates linear trajectories between the plane and points on A as potential waypoints, each further from the center of the obstacle than the last, until the projection of the obstacle center onto the potential trajectory is greater than the obstacles radius with a 10 meter buffer to ensure a collision-free flight path. Once an optimal waypoint is found to avoid collision with the obstacle on the original flight path, the potential waypoint is then run through a number of safety checks before being sent to the plane. The ground station first cycles through all the obstacles and checks that the waypoint is not being placed within any other obstacles. In the case that the waypoint is placed within an obstacle, the line A is recalculated such that the potential waypoints are being moved to the other side of the obstacle. We then check to see that the waypoint is not placed outside of the geofence. If that does occur, we recalculate in the same way, trying to avoid the obstacle by diverting the trajectory in the opposite direction. Once the potential waypoint has passed all safety checks, it is then sent to the ground control station as an auxiliary waypoint. This process runs every time the ground station receives new obstacle data from the interoperability server to adjust the flight path as the velocities of the plane and obstacles change. When the recalculated path changes, the ground station deletes the old auxiliary waypoint and replaces it with the new one.

### 2.4.4 Interoperability

The autopilot system has a redundant method to send data to the interoperability system at a minimum rate of 7hz. The primary system send autopilot data directly to the onboard computer, which then posts the data over our wireless network to the interoperability server. This allows for a steady 20hz link. This system has a backup over the RFD900+'s, which only allow sending data down at a varied rate from 7hz to 15hz. Therefore, no matter if we lose Wifi connectivity, we will ensure that we continue to reliably post interoperability data at the correct rate.

## 2.5 Software System

The distributed software system is based on web server technology. Below in figure 12 is a software diagram of the entire system.

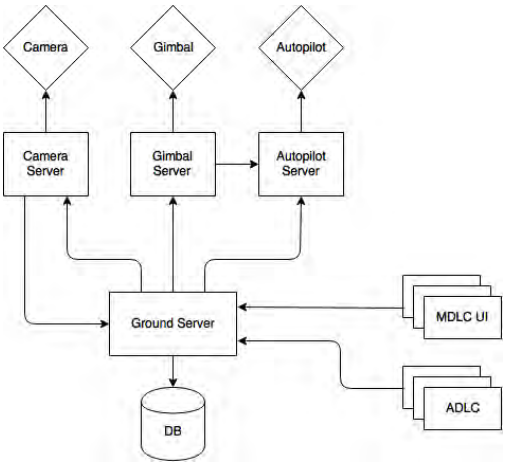


Figure 12: Software System Diagram

### 2.5.1 Air System

The onboard computer, an Intel NUC, runs both the camera and gimbal servers. It allows the imagery operator to toggle the capturing state of the camera as well as control the frame rate, shutter speed, brightness and gain settings of the camera. When the camera starts capturing images, the camera server spawns a new thread that grabs an image from the camera at the provided frame rate. It compresses the raw image to JPEG format and temporarily saves it locally as backup. Another thread handles sending the images down to the ground server. The gimbal server interfaces with the Simple BGC gimbal to ensure that the camera stays pointed at the ground even when the plane is tilted. It can be set to one of four modes: retract, pointing at ground, pointing at angles and pointing at coordinates. The gimbal server connects to the autopilot, which allows the gimbal to know the planes location and calculate the angle to the off axis target.

### 2.5.2 Ground Control System

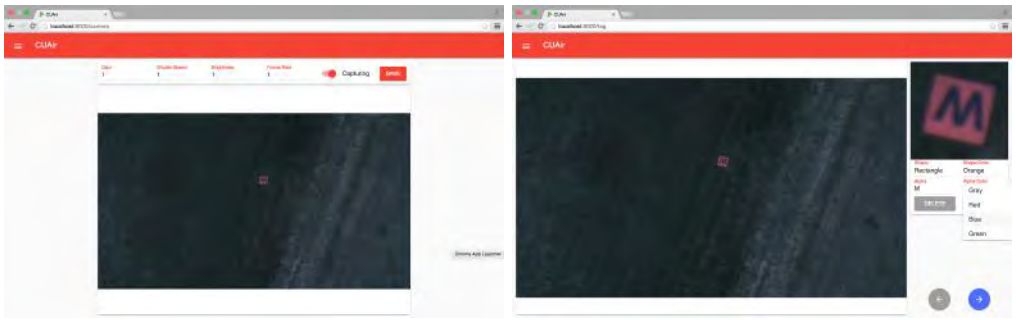

Figure 13: Software UI screenshots, Left: target tagging page, Right: camera settings,

The ground server interfaces with the servers on Theseus and provides a manual tagging user interface (UI). The tagging page of the UI allows manual taggers to request images to process, identify targets and input the identified characteristics. The software system ensures that images are evenly distributed over the set of manual and autononomous taggers and that every image is processed at least once by each system. An instance of a target spotted in an image is saved as a target sighting. The target sighting is geotagged using the telemetry and gimbal data corresponding to the image the target was spotted in. On the merging page of the UI, taggers can view all the identified target sightings and merge several sightings of the same target

into a target to be sent to the interoperability server. The settings pages of the UI allow the taggers to control the settings of the camera and the gimbal.

The geotag module of the ground server takes in the position of the target sighting in the image and the position of the plane and the gimbal at the time the image was taken to calculate the GPS location and heading of the target. This process is done by projecting a vector from the plane into the ground, with an error margin due to error in the roll and pitch of the aircraft. We do not need to take into account the roll and pitch of the aircraft since the gimbal ensures that the camera is always pointed to ground.

### 2.5.3 Distributed Networking System

In order to ensure that network disconnects between the plane and the ground do not impair performance, each component has reconnection logic. The camera server temporarily saves the captured images on the NUC and queues them to be sent to the ground. This ensures that no images are lost. Additionally, the ground server queues the gimbal/camera settings changes and retries those requests that timed out.

## 2.6 Autonomous Vision System

The Autonomous Detection Localization and Classification task was accomplished using Convolutional Neural Networks (CNNs). The team chose to utilize CNNs because of their success in a number of image classification challenges, as well as how easily they can be implemented using the Caffe library. As CNN's require a significant amount of data to train, the team developed software to create simulated camera imagery and therefore target sightings. Aside from curating data, the most important part of utilizing a CNN is deciding on the architecture of the network. The team tried a number of architectures that had been successful at other similar classification challenges. The team evaluated the architectures by training them on our datasets for two thousand iterations with a high learning momentum. A support vector machine was used for classifying shapes and a custom CNN was used for classifying shape colors. The other CNNs used the GoogLeNet architecture. The data was split at roughly 75% for training, 12% for validation, and 13% for testing. For the detection network our dataset contained roughly ten thousand images. Our dataset for feature classifiers contained roughly images. Testing was handled by a script that ran the trained networks on each of the test images, then automatically uploaded the results to a spreadsheet in Google Drive. This allowed the team to examine trained network's performance in detail and significantly helped fine-tuning. The target detector had a precision of 54% and a recall of 65%. The target feature classifiers had the accuracies shown in 14:

To estimate the overall performance of the vision system at competition the team developed a model that takes the accuracy of the individual components and generates the probability that the vision algorithm will correctly identify zero through 6 targets. The model simplifies the competition and the behavior of the vision algorithm to allow the competition to be simulated repeatedly. This simplified competition was simulated one million times while keeping track of how many targets had three or more features correctly classified.

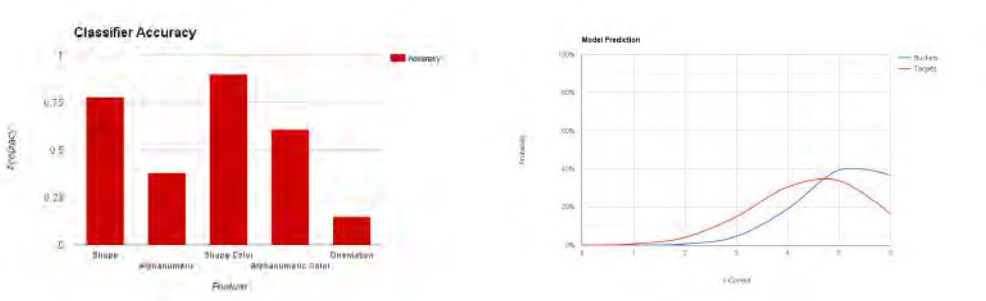


Figure 14: Left: Target characteristic classification accuracy, Right: Target recognition model accuracy

## 2.7 Data Links

The aircraft has four wireless links between the aircraft and the ground. The primary connection over which images from the machine vision camera travel is a point to point network between two Ubiquiti Rocket AC's. The link operates at 5.7 GHz using the 802.11ac Wi-Fi protocol. The autopilot system communicates between two RFD900+'s at 900 MHz. A Ubiquiti Bullet M2-HP is used to connect to the SRIC network which operates at 2.4 GHz using the 802.11g protocol. A final frequency hopping UHF link acts as the backup RC link, allowing for manual control of the aircraft from the ground.

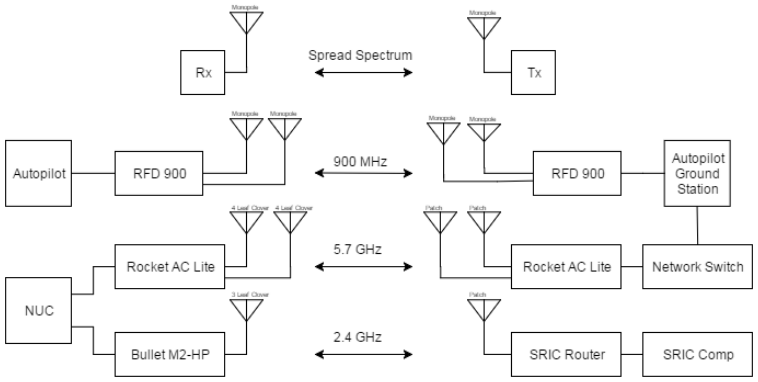


Figure 15: Communications systems diagram

In the air two 3 dBi skew-planar wheel antennas are attached to the Rocket AC over 4 feet of RG-58 coax cabling. These antennas were chosen because of their near omnidirectional radiation pattern and wide frequency band. This ensures that connectivity is independent of aircraft orientation. The antennas are placed roughly a meter apart, in each wing, to avoid possible interference with the carbon fiber boom and other electronics in the fuselage. The skew-planar wheel antenna radiation pattern results are shown above.

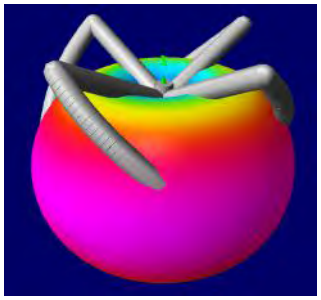

Figure 16: Skew-Planar Wheel Antenna Radiation Pattern

Two 20 dBi patch antennas are located on the ground. These antennas have a high directivity, and were chosen specifically to strengthen the signal coming from the aircraft. An estimated link budget is as follows in 3.

$$P_{out} = P_t + G_t - L_t - L_{fspl} + G_r - L_r \tag{3}$$

$$L_{fspl} = d_{dbm} + f_{dbm} + 32.44 \tag{4}$$

With the following parameters: transmitter output power of 27 dBm, transmitter antenna gain of 3dBi, transmitter loss due to transmission line of 1.5 dB, frequency of 5.7 GHz, distance of 1 Km, receiver gain of 20 dBi, and receiver and transmission loss of .75dB, the receiver power is -59.83 dBm. A -60 dBm received signal is well within the tolerance of the 802.11ac specifications and the well above the noise levels of the Rocket AC's receivers. A link budget calculation describes the received power of the data link.

## 2.8 Antenna Tracker

The half power beamwidth (HPBW) of the patch antennas for the data link on the ground is roughly 30 degrees. The radiation pattern is therefore highly directive with much lower sidelobes closer to the horizon. For this reason these antennas must always be facing the aircraft; this is the primary requirement

of the antenna tracker. The antenna tracker receives the aircraft"s GPS data off the autopilot server via a Raspberry Pi connected to the ground station. The Pi takes in sensor data about its GPS location and magnetic orientation and calculates the angles, both azimuth and altitude deflected from a set zero position, it must point to ensure it is facing the aircraft. This data is passed over serial UART to an Atmega microcontroller which is used to drive two DC motors to the correct position. Proportional feedback from the motor encoders is used the eliminate error and overshoot. Mechanically, the tracker was designed to be robust and easy to assemble in the field while also having the speed to track the aircraft. Field assembly is toolless with parts that come together using button connectors and clevis pins. The system consists of a main yoke in a U-shape that supports the outer antennas and the electrical hardware inside the frame. The frame rotates about the z-axis with a large central gear affixed to a tripod. The antennas use a worm gear to change pitch and track the aircraft vertically without slipping.

### 2.9 Launch System

Theseus uses a pneumatic catapult system, which CUAir has used for over two years. The pneumatic catapult launch system provides repeatable takeoff scenarios and reliable autonomous takeoff. The pneumatics were tested at over 150 psi - although the aircraft is launched at a maximum of 125psi. The catapult also has multiple safety mechanisms in place to ensure both Theseus and operator safety. First, a safety pin is installed to ensure the aircraft can't move if the catapult launches prematurely: this feature has been tested up to 150 PSI. In addition, there are software checks and a covered switch that ensure the catapult cant electrically launch unless safety criteria are met. The cart-launch system was kept simple to ensure ease of use and reliability. The aircraft sits cradled in the cart, with all of the weight resting on the cradle. The cart uses two pronged arms to support and push the aircraft forward. Once the cart and aircraft reach the end of the catapult, the aircrafs momentum folds the arms down and the plane is able to exit the cart and slide over the folded arms. Across all test flights, this system has only failed once - due to human error placing the plane on the cart.

## 3 Test and Evaluation

### 3.1 Stress Testing

All of the designs used in the fuselage were thoroughly tested to ensure reliability and flight worthiness. The hose clamps used to attach all components of the plane to the main boom were tested to ensure that no parts would slip and become misaligned during flight. An increasing torque was applied to a modified clamp until slippage was observed, resulting in an average torque on slippage of 124 N-cm. This is far above the expected roll torque created by the wings of 73.78 N-cm. The wings for the plane are attached to the main boom via a lasercut Birch wing plate. This plate was tested to withstand 378 N vertical lift force, which provides a factor of safety of 1.9 compared to the 200 N we expect the wings to produce during aggressive aerodynamic maneuvers. To determine the materials for our Fuselage ribs a number of different designs were tested to see which rib would be able to sustain and dissipate the energy from an impact with the ground. Since our plane lands without landing gear it was important that these ribs be strong enough to survive many landings while still being light. Thus the chart below was created. We decided on using a ply of four 3.1mm birch ribs due to their high strength/weight ratio and ease of manufacturing.
In addition to the individual component testing as mentioned in the airframe and fuselage sections above, the Theseus airframe as a whole was thoroughly tested while being flown. During flight, the roll, pitch and stall characteristics of the aircraft were tested using aggressive aerodynamic manuevers. The aircraft was flown overspeed and underspeed (stalled), and rolled 90° to the left and right. The aircraft recovered from each maneuver.

## 3.2 Autopilot System Performance

Basic flight data for competition aircraft: Thesues flew 26 autonomous and 32 manual flights. The aircraft completed a total autonomous flight time of 3:42:06 and a manual flight time of 3:52:19.

## 3.3 Material Considerations

When determining the manufacturing materials for the wings and tails, the team conducted several tests to determine if fiberglass, 3.1mm balsa, or 1.5mm balsa laminate over a polystyrene foam core would result in the lightest surface while still retaining structural integrity. The first test conducted was a simple mass test, as seen below in Figure 17

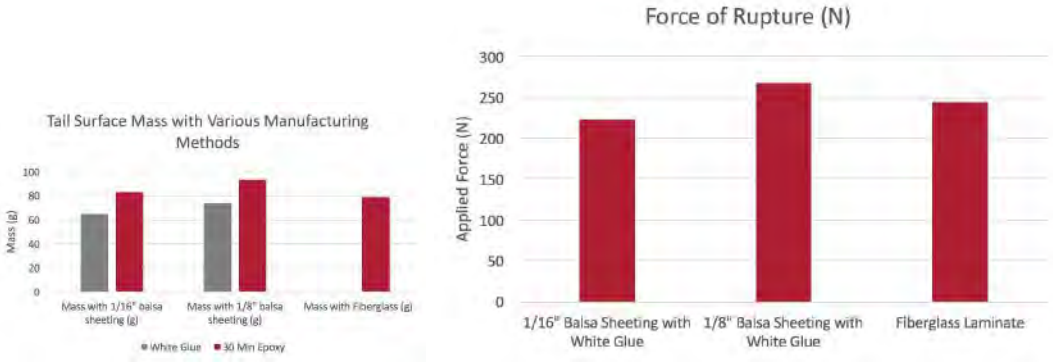


Figure 17: Left: Tail Surface Mass; Right: Tail Surface Strength

The team compared the strength of new manufacturing techniques to the previous year's fiberglass surfaces. Below is a setup of such a test:

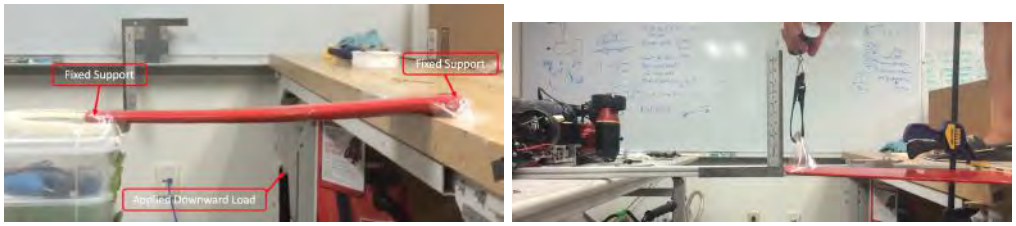


Figure 18: Left:Strength Test Setup; Right:1.5mm Laminate Test

From this table it was determined that the 1.5mm balsa specimen was only 9.1% weaker than the fiberglass specimen. In order to test the strength of the 1.5mm laminate for the tail, the tail surface was glued to a test tailbox, and a load was applied at the tail tip, by pulling on a luggage scale attached to the tail via packing tape. A picture of this test structure is depicted below.

From testing, the moment required to break the wooden inserts and 3D printed break-point was 9.33Nm, while the moment required to break the horizontal surface is 40.0Nm. Under normal conditions, the tail produces -2N of force, corresponding to a .28Nm moment about the tail, indicating that the connection point is strong enough to handle aggressive maneuvers while flying. Since the sum of the moments required to separate the stabilizer from the tailbox is much smaller than the moment required to break the stabilizer, the stabilizers do not break, but rather separate from the tailbox during a crash. Therefore, the 1.5mm balsa laminate specimen is strong enough to be used on the tail surfaces, and is 17.7% lighter than the fiberglass specimen. Using these materials, the airframe team manufactured four sets of tails, all with identical dimensions. The combined number of tails and modularity resulted in undisturbed testing of the

teams airframe and software. Since the team desired stronger wings than last year's fiberglass wings, they decided to use 3.1mm balsa laminate with polystyrene foam core. Two garolite strips were also attached on the top and bottom of each wing, which acted as an I-beam and greatly increased the stiffness of the wing. In the event of a crash, all the force is transferred from the wings to the wing plate via the garolite strips and the balsa laminate. This transmitted forces breaks the wing plate, which can be constructed in 2 days, instead of the wings, which takes one month to construct. The pod plane design also allows for interchangeability of components. Therefore a damaged wing can be replaced in under 10 minutes which allows for continuous flight testing. To test the strength of the wings, the team applied a 25 kg distributed load across the wings, as seen in the following figure. This load resulted in the wings flexing 50mm on either side, and corresponds to the wings experiencing 2.5 times its load during level flight. With a safety factor of at least 2.5, the team is confident that the wings are strong enough for even the most aggressive maneuvers. Below is a CAD rendering of wing cross section, where all the construction materials are shown and labeled.

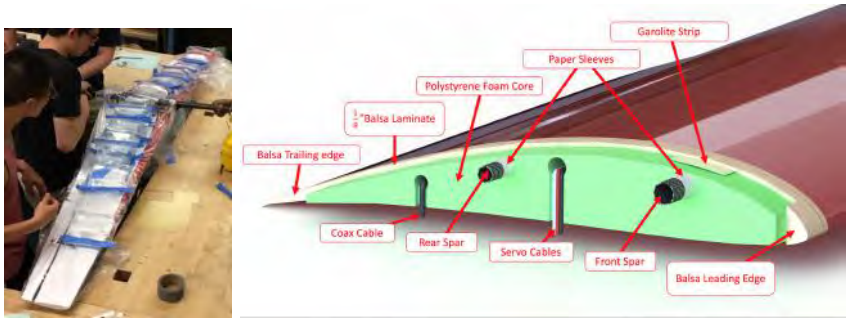


Figure 19: Left: Distributed load wing tests, Right: Cross section of wing

## 3.4 Payload Testing

Theseus's payloads went through extensive benchtop testing. Each sub-system was tested separately via rigorous unit tests. Once integrated, the system was also tested using mock autopilot data and moc -camera images. These tests enabled a complete check of system performance as if it were in the air. Theseus's payloads were also tested in flight under both manual or autonomous control. This allowed testing of the gimbal system, camera system and antenna tracker. The tests were conducted over a range of different weather conditions.

## 3.5 Communications Data Link Testing

Throughput of the data link was tested between two points 650m apart with line of sight. A variety of frequencies, antennas, and wireless modules were tested using a static test configuration over this distance. Mock-data was transferred and the data rate measurements were taken over a one minute period. Below in 20 is the plot of the best results from this test. Point-to-point Rocket AC's operating at 5.7 GHz are used with skew-planar wheel and patch antennas. It provided the best data rate of roughly 11.5 MBps.

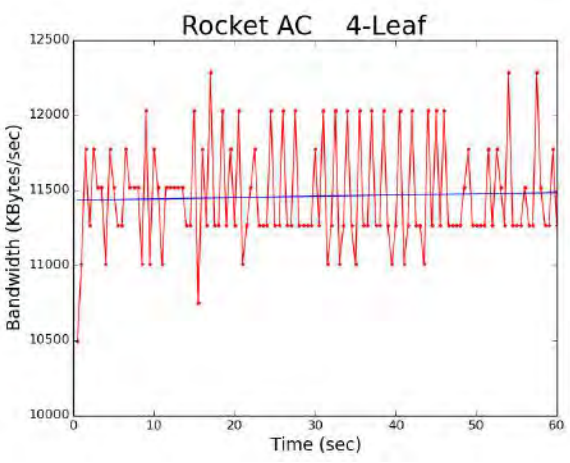


Figure 20: Distance Testing Results. Best test results plotted

### 3.6 Geotag Testing

For each test of the geotag algorithm, we tagged 35 unique targets and found that we were able to tag 22 of the 35 targets to 50ft accuracy and all of the targets to 250ft accuracy.

## 4 Safety Considerations

### 4.1 Design Safety

The primary concern while designing the aircraft was to ensure that all systems would fail cleanly and safely, and that systems would quickly recover from failures. CUAir designed a very large factor of safety into each system - from the high 1.5 factor of safety for all fuselage shocks to a failure of less than 1MB/sec of data transfer rate for our wireless link. The team also tested all systems at extremes to ensure that the aircraft could safely takeoff, perform the mission and land in any competition condition. Lastly, failsafes, from catapult pressure failure to aircraft throttle failure, were tested. CUAirs hardware system contains failure points, to ensure that shocks of landing or crashing won'o harm the electronics or sensitive airframe components. All components of the software system contain reconnection logic to ensure that the images, as well as corresponding telemetry and gimbal data, are transmitted reliably.

### 4.2 Operation Safety

Our operating procedure follows strict Go/No go flying based on a predetermined checklist. The operators step through the checklist before every launch to ensure repeatable and reliable flights. Lastly, test flights are always attended by at least one person with first aid training. The team is very cautious about personal safety - especially since Theseus utilizes high voltage LiPo batteries for flight. All team members are trained to use a fire extinguisher in case of a LiPo fire or overheated components.

### 4.3 Risks and Mitigations

There are multiple safety risks when flying a large RC aircraft utilizing a pneumatic catapult launch. The safety considerations are split into three categories: operators, specators and aircraft. The team's flightline keeps spectators safe by preventing them from entering the active runway or flight area. Each operator has a unique role, ensuring that team members are only located near the aircraft or catapult if necessary. The catapult has hardware safety features to ensure that an operator is safe to work near the catapult even if it's

pressurized. Additionally, the switch panel has a motor-disconnect switch, allowing an operator to quickly disable the motor if needed. There are several methods to keep Theseus safe. First, and foremost, we have the ability to take over manual control if something goes wrong on the airframe. Second, the airframe is designed for point-of-failure modes, so even in the worst case scenario crashes, expensive components will be saved.

# 5 Conclusion

This paper summarizes the work done by CUAir in preparation for the AUVSI SUAS 2016 competitions. The CUAir team has dedicated itself to designing, building, and testing the Theseus UAS. This year involved a completed aircraft redesign, software rewrite and substantial payloads overhaul. New work was done on advanced target detection algorithms. Extensive testing has been performed on the entire system to ensure the UAS is reliable and that meets all of the specifications laid out in this paper.

# 6 Appendix

## 6.1 Cyber Security

The CUAir team has thoroughly secured both the ground station and the vehicle system. The autopilot system communicates with the ground station over an encrypted data link. The ground station connects to the autopilot over both the RFD900 link and the wifi link to ensure that if either is jammed or goes down, the system will always be able to connect, interact with the autopilot and receive interop data. The transmitter utilizes 433Hz frequency hopping and therefore not be jammed. The distributed system communicates with the plane and sends images over a WPA2 secured network. Additionally, the autopilot ground station is password protected such that no POST requests can be made without full authentication, however data can still be viewed without authentication for the convenience of the judges. All of these measures were implemented to ensure that no one could take over control of the aircraft or the ground station. We protect against GPS spoofing by changing from GPS to dead reckoning if the GPS drifts beyond normal values. This was chosen as dual GPS do not work to overcome GPS spoofing on the ardupilot system.

## 6.2 Flight Item Checklist

# PRE-FLIGHT

CU AIR
Cornell University Unmanned Air Systems

### ☐ Stage 1 - Per Day

**Precondition:**
Plane and 6 boxes are on tarps
Boxes are labeled on page 1 with *

**Postcondition:**
Plane is CG'ed & Calibrated

- Wings on & plugged in
- I2C plugged in
- clover antennas plugged in
- Check battery levels
- Autopilot battery plugged in
- **THROTTLE NOT PLUGGED IN**
- Surface deflections correct (man/stblz)
- Surfaces are trimmed
- Receiving plane vitals
  - (GPS, airspeed, compass, roll, pitch, yaw)
- Compass calibration
- Plane Dance (rename this)
- Stabilize works
- Plane CG test
- Wing tip test
- Range Test
- Xbee Range Test

### ☐ Stage 2 - Per Battery

**Precondition:**
CG'ed, calibrated, & on Catapult

**Postcondition:**
Surfaces & Throttle Tested, Range Tested. On Catapult

- Put on Catapult
- Lift Test
- Test all Surfaces
- Plug in Throttle Battery
- Throttle test
  - Check throttle is 100% on Ground station
- Throttle correct direction
- Put nose cone on
- Redo barometer calibration
- Redo airspeed calibration

### ☐ Pre-Launch

**Precondition:**
On Catapult, Everything tested

**Postcondition:**
Surfaces and throttle checked again. Catapult pressurized

- Check servo battery
- Catapult will not clip box
- No slack in catapult line
- People:
  - Pilot & Spotter
  - 2 on Catapult
  - Timer & Video
- Check plane won't lift off catapult prematurely
- Transmitter on HIGH
- Pressurize Catapult
  - Odysseus: 115 psi
  - Bixler: 35 - 45 psi
- Surface Test
- Throttle Test
- Safety pin
- Capturing images
- Point gimbal at ground

# LAUNCH

- Catapult is removed from the runway
- Flight timer report every 30 seconds

# UPON LANDING

### ☐ Post-Flight Directions

- Transmitter is on
- Throttle is turned off
- Only when the plane is off can the transmitter be turned off
- Check for damage on the plane
- Create interop landing event for flight