Fall | 16

Maryland UAS Team

The Maryland UAS Tea is competing in the AUVSI Student UAS Competition for the second time; this system is based on the RMRC Anaconda Airframe and Pixhawk autopilot system. Comprised of 7 flight line operators and a safety pilot, this team plans on completing the Flight Readiness Review and Journal Paper. Unfortunately, due to new FAA regulations and poor timing with flight partners, the team was unable to fly for the spring semester and therefore was unable to record a proof of flight.

TABLE OF CONTENTS

SYSTEMS ENGINEERING APPROACH

MISSION REQUIREMENTS ANALYSIS

The systematic approach the Maryland UAS team took to create Shell Shock can be divided into two parts: (1) understanding competition constraints regarding system characteristics, environmental constraints, mission tasks, safety, and competition layout and (2) designing, building, testing and unmanned aerial system capable of meeting these constraints. Having never competed in the competition before, the Maryland UAS team had to first focus on competition layout and mission task constraints to gain an understanding of what generally the system needed to be capable of and how its success could generate points in competition. From this understanding, the team could develop a vision for what the mission of the system would be and how they would make it a reality using the function, design, safety, and competition procedural constraints as a basis for all design, construction, and verification. Originally, the team's goal was to accomplish as much as possible starting with the primary tasks and building up. The team had high aspirations and put little off the table. The work was largely mission task requirement focused, and it was divided amongst four sub-teams: autopilot, communication systems, imaging, and mechanical systems. In early stages of system design, each sub-team utilized past competition journal descriptions of successful systems. Being the team's first year, the design was to be made up of established hardware and software and the focus was to be on integration of these parts rather than part design.

As the sub-teams started building their respective subsystems, they gained an increasingly better understanding of the different competition constraints and continually improved their processes. For starters, the sub-teams were re-organized into imaging, software, mechanical, and electrical. Beyond that though, as design and construction progressed, the safety and systems requirements became increasingly important to keep in mind as systems requirements are the glue that holds the aircraft system together and safety requirements help bridge the transition between design, construction, testing, and verification (and compliance with them are necessary for the plane to leave the ground). The team's leadership implemented a continual feedback loop of adjusting competition mission goals based on team progress and timeline/Gantt analysis. Because timelines and work breakdowns could not be based on prior experience, the team's leadership continually assessed what could actually be accomplished based on competition timeline and sub-team progress and interdependencies. Once in the construction and testing phases, internal timelines were constructed based on safety. It was crucial that each part of the system was safely constructed, inspected, and tested. The team put safety in the forefront of construction, inspection, and testing by ensuring that all systems met competition safety requirements and by completing a COA.

In creating and implementing a systematic approach to ensure that all necessary requirements were met, the team found that how the requirements were presented to the subteams was just as important as making sure that all of them were compiled, met, and verified correctly. At first, the requirements were compiled in a spreadsheet that attributed each requirement's section in the competition rules, which subteam(s) the requirement applied to, how they would be verified, which testing iteration of the plane they would belong to, and what each sub-team needed to do and needed from other sub-teams to make see the requirement come to fruition. At the end of the day, this spreadsheet was converted into a checklist of requirements by task (and in general) that sub-teams could more easily use.

Design Rationale

Airframe

The RMRC Anaconda was chosen due to its simplicity in setup and the built-in capability to carry a variety of payload items, such as FPV and belly-mounted cameras. Since the airframe was purchased as an Almost Ready to Fly (ARF) PNP kit, it only required the team to set up the power system and chosen payloads, reducing the amount of time necessary for setup and increasing the amount of time the team had to conduct test flights. Competition requires that the UAS be able to fly for extended periods of time along with having the capability to take images. The Anaconda airframe meets these requirements. Modifications to the airframe include openings cut into the floor of the fuselage for camera's, antennas, and other sensors. These modifications will be examined in further detail in the system overview.

Autopilot and Mission Planner

The Pixhawk system is the plane's autopilot. We chose this system for a variety of reasons, most notably because of its large, open source development community, and its price range (sub $500).

Mission Planner, the ground control software we chose to use, is also an open source platform that happens to be free. Those two factors, as well as our team members' familiarity with the program, were the primary reasons that this program was chosen. With the open source nature of the program, the team felt that Mission Planner would be a much more long-term system to maintain since we would be able to continuously update the program as the competition evolves.

Communications

We are using the 5GHz Ubiquiti Bullets and Nano-stations to communicate between the aircraft and the ground station as opposed to the 2.4GHz Ubiquiti Bullets and Nano-stations used last year. This switch was because we are using an RC controller that communicates over 2.4GHz. We are also changing the antenna used on the aircraft from the omnidirectional stick antenna to an omnidirectional cloverleaf antenna that will also make up for the range decrease occurring by moving frequencies from 2.4 to 5GHz. The Nano-stations will be attached to a simple system that will be directed at the vehicle by human movement.

The interoperability system consists of two Nano-stations broadcasting at 5GHz supplied by a router. The link from the bullet to the Nanostation is secured via a WPS2 encryption key as is the router that the Nanostation piggybacks off of. Connecting to this network, the bullet then transmits data back to the computer through the script operator on mission planner. The data is then transmitted through a python script running in the command prompt of the computer to the server running on a desktop that is a mockup of the judge's setup at the competition. This data is then uploaded to the AUVSI site through the username testuser and the password testpass.

Payload

The camera chosen for our system is the Nikon J4 mirror-less DSLR with an 18.5mm fixed lens. This camera was the final choice due to its lightweight build, overall small size, and ability to exchange lenses for optimal image capturing. The team wanted a camera that took relatively high-quality images to be able to accomplish automatic image recognition in the future and also to be able to zoom in on the image without losing a large amount of resolution. We also wanted a camera that had a manual mode since autofocusing during flight would result in a delay in taking pictures, which could potentially lead to missed targets. The lens was picked because of its fixed focal length so that the camera would not try and auto-focus when taking a capture

Gphoto2 is a powerful command line interface (CLI) tool that allows remote control of a camera tethered to the system running it. Using this interface allows us to write scripts to control the operation of the camera. These scripts are stored on the Raspberry Pi 2 and are called by the ground control station while connected to it over the network. Gphoto2 also provides us with the ability to not only control the image capture but also the download the images from the camera to the Raspberry Pi 2 while in flight. These images are then placed into a folder, which is synced with a folder on the ground control station. There are other programs that provide a similar functionality, however they do not make use of a CLI, creating difficulties when trying to run it while SSH'ed into the Raspberry Pi 2.

BitTorrent Sync is commercial syncing service that creates a system of folders across all our devices that are continually synced with each other while connected to a network. This service is run on the Raspberry Pi 2 and our ground control station systems. When an image is downloaded from gphoto2, it downloads to the current directory at the time of download. Using Sync allows this folder to be synced across all approved devices on our network. So when an image is captured and downloaded, seconds later it appears in a folder on our ground control station. One big advantage of Sync is its security. In order for folders to be synced, a key or link is used to approve it on the second device, otherwise the syncing does not occur. Prior to choosing Sync we experimented with scripts to secure-copy the images from the Raspberry Pi 2 to the ground control station. However, this meant that all the images would need to be taken in order to copy them over and we would be connecting/disconnecting/reconnecting multiple times thought the flight. In an effort to find a better solution we found a program called Syncthing that provides a similar service. However, a few weeks into using this service, devices were experiencing problems connecting to each other. This prompted us to find another solution, leading us to BitTorrent Sync. In addition, we found that Sync is quicker and more efficient than using secure-copy.

Expected Task Performance

This year's system was built with the intention of completing autonomous takeoff and landing, autonomous navigation, and image capture and transmission for the Search Area and Actionable Intelligence tasks. The flight and search area tasks were chosen since they are required of the system prior to being able to complete further secondary tasks. These tasks have been the main focus of our flight tests so we have a high confidence in their completion. Since our system is capable of transmitting imagery from the air, we are also going to pursue the Actionable Intelligence task as our first secondary task. The imagery ground control team will do the target detection manually. With the test images that we have been able to obtain from the camera, we are also confident in our ability to complete this task as well.

Programmatic Risks and Mitigation Methods

We initially tried to implement a formal risk management system using a risk matrix and mitigation plan. However, this proved too cumbersome for our application. In order to simplify things we divided risks into two categories. The first category was "showstoppers". These were risks that had a reasonable chance of preventing completion of a mission task or worse. Any showstopper was mitigated immediately before any further flight-testing. The next category was "projects." These were issues that would not prevent flight, but would make it easier to complete existing tasks or complete additional ones.

SYSTEM OVERVIEW

AIRFRAME

For the competition, the MUAS airframe needed to meet certain specifications based on design preferences and competition requirements. To produce a successful system, several key requirements were considered: easy modification, quick build time, and affordability. As a team, we decided that the RMRC Anaconda, as shown below in Figure 1, was the aircraft that met these specifications.

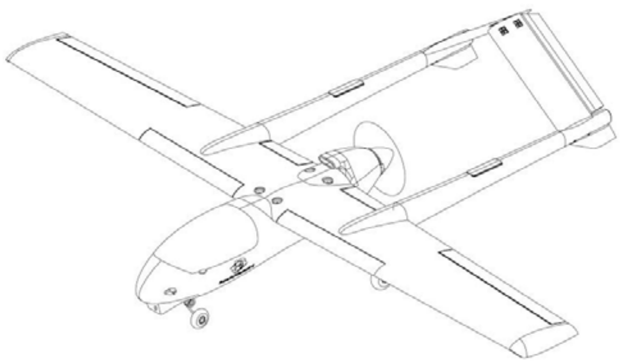

FIGURE 1 - RMRC ANACONDA AIRFRAME

FIGURE 2 - ADJUSTABLE FUSELAGE TRAY

The RMRC Anaconda was purchased as a “Plug and Play” (PNP) kit, which includes an installed motor, esc, and servos. The Anaconda is an inverted-V tail pusher design geared towards aerial photography and FPV. The Anaconda has an approximate maximum flight time of 45 minutes, which means the plane can stay in the air for the entire flight window of competition. Due to its lightweight design, functional flaps and leading edge slats, and large wing area, the Anaconda has excellent stall characteristics that allow for easy recovery of the airframe from stall, as well as short takeoffs and landings. The molded bead foam construction of the airframe also allows the team to easily modify the airframe. The foam construction also adds some degree of impact resistance, and is relatively easy to make minor repairs to.

The fuselage cavity has dimensions of 26.5in x 6in x 3in. This allows for ample room to store all of the internal components including the PNP components with the added MUAS components including a camera, Raspberry Pi 2, Pixhawk, GPS module, and Ubiquiti Bullet.

Along with that, the fuselage contains an adjustable center of gravity tray, shown in Figure 2. This helps with compartmentalizing and organizing the electrical components as well as making changes to the center of gravity to make sure the UAV is balanced. The tray has the ability to move two inches, which is more than enough to fine-tune the location of the CG, which should be set at 3.12 in. from the leading edge of the wings according to manufacturer specifications. Additionally, the booms of the plane contain 3 contact audio jacks that connect the V-tail servos to the fuselage of the plane for ease of connection during setup.

Below are the dimensions and specifications of the major mechanical components of the plane. These components are required for flight, either RC or autonomous, and therefore make up the backbone of the system.

Wingspan	81.10 in.
Wing Area	759.5 in.^{2}
Length	55.5 in.
Flying Weight	10lbs
Propeller	APC 15x4E
Servos	3x30g, 2x17g, 2x16.44g metal gear digital
Battery	2x Li-Po 4s 5100 mAh
Motor	RMRC Tiger Motor Brushless Outrunner AT3520-5, 800 kv
ESC	RMRC Tiger Motor 80A ESC with switch mode BEC and 10A Castle Creations BEC

TABLE 1 – AIRFRAME PARTS AND SPECIFICATIONS

MODIFICATIONS

For imaging capabilities, a downward facing camera needed to be mounted in the Anaconda. Due to center of gravity constraints, the camera had to be mounted as far back in the fuselage as possible. A hole for the lens was cut in the bottom of the fuselage, and the camera was mounted just above on a vibration absorbing foam pad. As shown below in *Figure 3*, the camera interfaces with the foam with industrial-strength Velcro while the foam then interfaces with the fuselage bottom with Velcro again. This setup allows for security while dissipating the vibrations reaching the camera, as the foam acts as a spring-damper system. The hole in the fuselage is 2.3 inches in diameter, 8.5 in. from the back of the plane.

FIGURE 4 - CAMERA SHOCK-ABSORBING SPONGE

FIGURE 5 – SENSOR AND ANTENNA POSITIONS

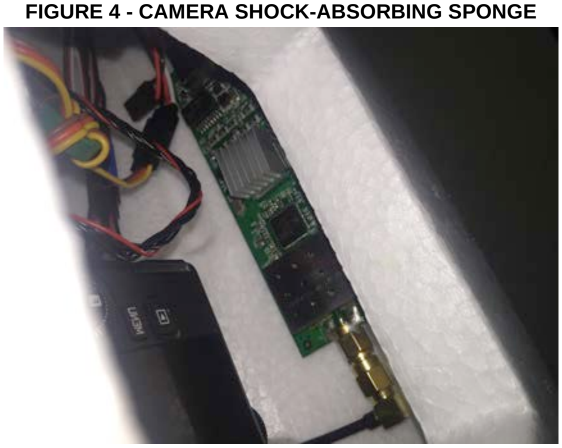

FIGURE 6 – UBIQUITY BULLET POSITION

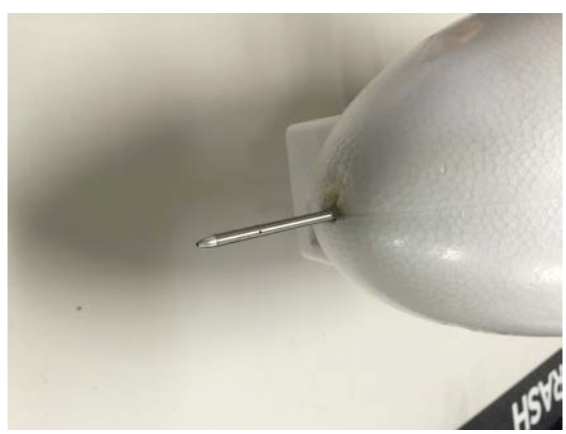

FIGURE 7 - PITOT TUBE MOUNT

Due to the overall suitability of the Anaconda design straight off the shelf, not much else had to be modified. Holes were made in the bottom of the fuselage for the antennas of the 915mhz telemetry link, the circularly polarized antenna for the Ubiquity bullet, and the Lidar-Lite laser rangefinder. A pitot tube was mounted in the very front of the fuselage to measure airspeed.

PROPULSION SYSTEM

As mentioned before, the Anaconda is an electric plane that uses the RMRC Tiger Motor Brushless Outrunner AT3520-5 and an APC 15x4E propeller. The motor is coupled with an 80-Amp RMRC Tiger Motor electronic speed controller (ESC) and two RMRC 4S 5100mAh lithium polymer (Li-Po) batteries connected in parallel.

AUTOPILOT

The Pixhawk system is the plane's autopilot. This is an open-source autopilot which allowed our team to make a number of changes very simply as opposed to other autopilots that were researched. Included in the system are GPS, airspeed sensor, gyroscope, ppm encoder, compass, accelerometer, magnetometer, current sensor, and barometer. These sensors allow the plane know where it is and what it is doing, making it safe and allowing it to execute missions reliably.

Among the Pixhawk's various capabilities is the ability to update the mission in air. As a result, we are able to change the plane's path remotely, while it is in the air. This not only helps us at competition, but also adds another layer of control to the plane, making it safer to fly.

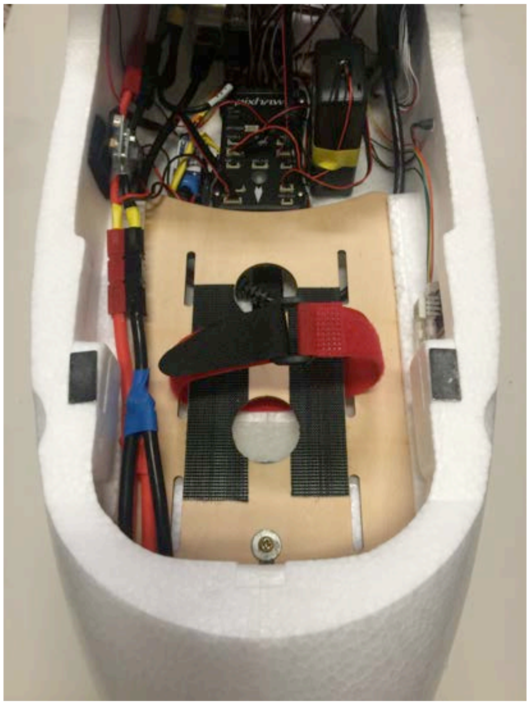

FIGURE 8 – PIXHAWK POSITION IN AIRFRAME

Mission Planner

Mission Planner is the Pixhawk's control software, and allows us a remote link to view everything related to the Pixhawk. Like the Pixhawk, Mission Planner is also open source, and has a large community surrounding it, which helped us in understanding various features of the software, as well as troubleshooting some issues we found while testing.

Mission Planner has numerous features. It has a 915MHz radio link to the Pixhawk, allowing us to see real time information of the plane, including the battery charge level, the position and orientation of the plane and more. Mission Planner fully supports all of the Pixhawk's abilities, which means we are able to use it to update the mission mid-flight, making our system as flexible and robust as possible.

Perhaps one of the most important features of Mission Planner is the ability to set various fail-safes. Fail-safes add an extra layer of safety to the system and allows the Pixhawk to handle events, which would otherwise be disastrous, to be handled as safely as possible. Without this feature, competition officials would not allow us to compete at competition. Mission Planner allows the Pixhawk to control the plane safely in the event of low battery, a link drop to the pilot or Mission Planner computer, GPS loss, and avoiding a no-fly area. If any of these events occur, we can set the plane to loiter, fly in circles, return to launch, or in a worst-case scenario, spiral into the ground.

Electronics

For the electrical subsystem, the most important task was to make sure that all necessary components of the vehicle received an appropriate amount of power and current. If the electrical system des not function properly, then none of the other teams could be expected to have their systems function either. Based on the selection of motor by the mechanical team, a 4S (14.8V battery) was selected. This was the lowest battery rating that would give the motor the power required to generate the lift necessary for flight. We chose Ready-Made RC 5100mAh 35C battery packs. Two batteries are used in parallel to achieve proper weight and balance, as well as longer flight times.

To convert the voltage from the battery down to a level that the Pixhawk is capable of handling, we are using an APM power module that is designed for compatibility with the Pixhawk. The APM power module converts the voltage from the battery into a maximum 5.3V to power the Pixhawk. One end of the power module is connected to the battery, the unregulated output goes to the ESC, and the regulated output is directed to the Pixhawk.

The electrical system also includes an ESC (electronic speed controller). The main purpose of this addition is to convert the DC power from the battery into modulated AC current to power the motor. We elected to use the T80a ESC included with the airframe PNP kit, based on reviews as well as recommendations from our pilot, who has significant experience building aerial vehicles. The input for the ESC comes from the non-regulated output of the APM power module.

In the event of some kind of power failure, our goal is to be able to bring the vehicle down safely. To implement this, we added a Castle Creation UBEC (battery eliminator circuit). The ESC has a built in battery eliminator circuit to supply 5v power, and would normally be used to power the servos and RC equipment in a normal manual control only build. However, this can create a dangerous single point of failure. ESC's are one of the highest stressed components in the plane, and if it fails in a normal configuration, not only does the aircraft loose power to the motor, there is also no control of the airframe anymore as it also powers the rest of the control equipment on the plane. For this reason we decided to use the BEC in the ESC to power only the Raspberry Pi 2, and power the servos and with the Castle Creations UBEC. This way, if the ESC fails, we will still have control of the aircraft, and can complete an emergency dead stick landing. In addition, the Pixhawk can draw power from the UBEC connected to its servo rail, as well as from the power module, creating a redundant power delivery system for the autopilot.

With our use of the Pixhawk, we are also using a number of included modules, included the airspeed sensor, GPS, master switch, buzzer, and LED. The airspeed sensor and GPS are use by the software team for location purposes, and the master switch, buzzer, and LED are all used in the powering up of the system to let us know whether it has been successful or not. All of these have special ports on the Pixhawk and are powered by the Pixhawk, so there was no extra work done to include them.

For telemetry, we chose to use the jD-RF900Plus Longrange telemetry system. This system has several advantages over the telemetry system made by 3DR for the Pixhawk. The RF900 is a 1W power system, with dual antennas, whereas the default telemetry system is a 200mW single antenna system. This increase in power and signal diversity allows for much greater range and reliability. The 3DR telemetry system is rated to 2km, whereas the RF900 has been shown to work to over 20km. The RF900 also includes built in real-time AES encryption, allowing for a more secure connection.

For manual control of the UAV, we use a Spektrum DX-7 transmitter and AR8000 receiver combination. This transmitter and receiver combination use frequency hopping spread spectrum technology to minimize the impact of interference on any given channel in the 2.4GHz spectrum. The receiver is a system of two receivers with multiple antennas to attempt to minimize the impact of RF dead zones in the plane caused by the signal reflecting of components. Additionally the system uses Spektrum's DSM-X communication protocol, which is designed specifically for environments high in 2.4GHz interference. Because of all these factors, we believe this is the best manual control system for our application, allowing for the plane to be recovered manually if the autopilot attempts to control the plane incorrectly or in an unsafe manner during flights.

PAYLOAD

The imaging system operates on a 2.4GHz Wi-Fi local area network, powered by a TP Link WDR4300 a/b/g/n Router, as well as a LAN between the computers at the ground control station. The router generates a signal with data transfer rates of up to 300Mbps, allowing for quick image and data transfer rates between computers. The LAN allows for all three ground control station computers to quickly images and data between each other for seamless operation.

In order to extend the range of our Wi-Fi network, an Ubiquiti Nanostation and Ubiquiti Bullet were purchased. The Nanostation is capable of transmitting 150Mbps across distances of at least 13km and will be located at the ground control station. The Bullet is rated for 100+Mbps over approximately 50km, depending on the antenna. To save weight and allow for the installation of a different antenna, the bullet main circuit board was removed from its weather proof plastic housing, and the large heavy antenna connector de-soldered from the from the front of the board. A much lighter connecter that interfaces with VAS - 5.8 GHz Cloverleaf Whip Antenna (RHCP) we decided to use was the soldered back on. This antenna is more suited for aerial applications as it provides equal signal coverage to a 360-degree sphere, instead of a cone or plane as more common directional and linear antennas provide.

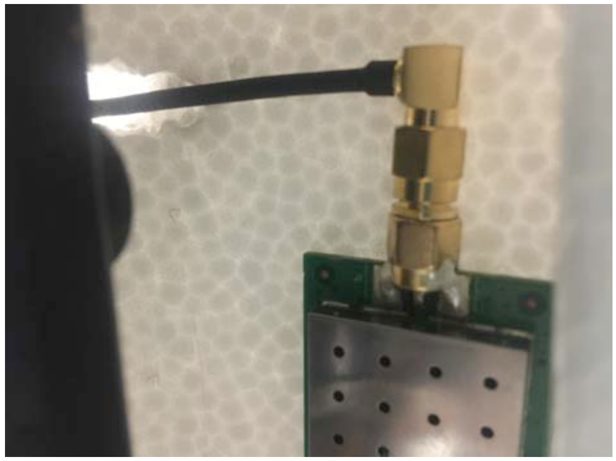

FIGURE 9 – BULLET CONNECTOR MODIFICATION

Connected to the Ubiquity bullet on the plane via an Ethernet cable is a Raspberry Pi 2 single-board computer that controls the camera and data link. The Raspberry Pi 2 runs the lubuntu operating system as well as the gphoto2 command line interface, which allows the ground control station to remotely operate the camera from either a command line script or through a Secure Shell (SSH) connection to the ground control station.

The camera that was chosen for the system was the Nikon 1 J4 mirror-less DSLR camera. This camera takes 18.4MP images, and the body weighs only 6.8oz, compared to other DSLR's at a similar quality coming in at 18oz before a lens. Along with this body, the 1NIKKOR 18.5mm f/1.8 lens was purchased to replace the default 10-30mm adjustable lens. This lens has a 47-degree field of view, allowing us to see a large portion of the ground below the plane without distortion.

Once the camera captures the photos, they are downloaded to a specific folder on the ODroid that is synced using a tool called BitTorrent Sync. Sync is a commercial syncing tool that allows a folder or folders to be shared among a group of computers that are connected to the same network. This ensures that when our photos are taken, they are immediately synced to the ground control station over the Wi-Fi connection for immediate processing and classification.

COMMUNICATION PROTOCOLS

The imagery system uses a 5.8GHz 802.11b/g/n Wi-Fi network between Raspberry Pi 2 and GCS using Nanostation, TP Link, and Bullet. This link was secured using WPA2 security to ensure that it met the competition standards. The autopilot communicated on a 915MHZ link with one of the ground control station computers. The RC Controller operates at the 2.4GHz band. The controller uses DSM2 to communicate with the AR7000 receiver and avoid radio interference from other transmitters to maintain full control of the aircraft at all times.

Testing and Evaluation

Mission Task Performance

The team was able to do a number of RC flight tests to verify our system's performance. Our safety pilot conducted these flights at the Free State RC field. These flights began with sandbagged RC flights with the minimum flight systems. After these were proven successful, systems and capabilities were added to flights to verify those systems. These included: the addition of the Pixhawk for telemetry data, the RC control of the plane through the Pixhawk, RC flight through the Pixhawk with fail-safes activated, RC flight through Pixhawk with the imagery system running, RC flight through Pixhawk with fail-safes active and imagery system running, future autonomous flight with sandbags, and future full system flights. This sequence of events is detailed in Appendix 1: Testing Schedule.

Payload System Performance

The payload system went through a large number of range and speed tests in order to verify that the team would be able to use the system to compete. The goal of these range tests was to reach 2500ft with a low level of transmission speed loss. At the moment, the team has only been able to test the imagery system up to 1200ft due to line of sight constraints at available spaces. Other issues with testing included our portable power system, which provided 12V instead of 24V to the ground control station antenna. After the purchase of a generator, the team plans to retest the range of the imagery system to accomplish competition ranges. If the imagery system is unable to attain the full goal range, the team is still able to compete and acquire imagery data. The limitation will be the frequency in which the data will be downloaded to the GCS.

After the range tests were completed to the best of the team's ability, the system was then tested on the plane for image quality. These were very subjective tests where the goal was for team members to be able to identify the test target characteristics from the images. At an altitude of 300ft and a ground speed of 35mph, the team was able to clearly see all images of test targets when the plane was in level flight.

Autopilot system performance

In order to be sure that everything was as safe as possible, and to maximize our chances of success, we preformed extensive testing before attempting to fly autonomously. First, we loaded the rover firmware to the Pixhawk and tested waypoints and pilot controls on a rover. This allowed us to familiarize ourselves with Mission Planner and the Pixhawk in a safe and controlled environment, without having to worry about damaging the plane or any of the safety concerns of operating the plane.

Next, we removed the propeller, loaded the plane firmware, and set waypoints using Mission Planner. This allowed us to walk the plane to each waypoint and double check that the control surfaces were moving in the correct way to move the plane to hit the waypoints. In addition, we performed fail-safe tests on the ground, manually triggering each fail-safe and verifying that the plane reacted as it was configured. We also preformed range testing of the 915MHz link from Mission Planner to the autopilot, in order to check the max distance that we could fly before experiencing problems with the link.

Interoperability system performance

The testing of the interoperability system was conducted by using a laptop equipped with both mission planner and python as the access point for the vehicle and a desktop as the judge's computer. The Nanostations and bullet were connected over one router on the 5GHz frequency, the Nanostations via

Ethernet and the bullet through the Nanostations. Separately, the two computers were connected via a different router both through Ethernet. The telemetry data was transmitted by the bullet back to mission planner on the laptop. Through the second router, the data was sent to the desktop, which uploads the data to the server on the AUVSI site.

Support for Likely Mission Accomplishment

So far, the team has successfully verified all capabilities of the aircraft and ground control systems up to autonomous flight. Many of these systems have had greater than ten successful test flights and all systems have had greater than 5 successful ground runs. The team feels very confident in their ability to successfully complete the portions of the competition listed above, including autonomous flight after testing, due to the testing and verification processes listed above.

Safety Approach

Safety criteria for operations and design

Both mission and safety requirements were at the forefront of the design, creation, and testing phases. Although mission requirements are undoubtedly important, the plane's ability to ultimately pursue mission requirements depends heavily on the ability to meet safety requirements. Safety requirements are flight critical requirements that demand safe mission completion. That is, even a failure of the system's critical systems, the plane will have the ability to terminate its flight without damage to other property or human life. The majority of the team's safety requirements were either derived from the competition rules or from risk analysis. In addition, the team pilot's prior flying experience added depth to the team's understanding of both the nature and importance of safety requirements.

From the competition rulebook, the team primarily used the safety requirements section and the pre-flight checklist as primary sources for safety criteria. The safety requirements section primarily provided environmental requirements, failsafe requirements, and safety-pilot requirements. The team identified the provided pre-flight checklist as a means for ensuring flight-critical systems are properly functioning and components of the system are properly secured for flight prior to takeoff. During the design phase, these competition safety requirements were utilized in choosing the system components discussed earlier. During the construction and testing phases, the pre-flight checklist was largely utilized to ensure system components were properly assembled and secured for safe flight.

In addition to the competition rulebook, internal risk analysis and lessons from the team pilot yielded insight with respect to criteria for meeting these requirements. The biggest decision stemming from these discussions was the use of a pre-flight checklist. Prior to first flight, each sub-team made a list of the final things that need to be checked prior to take off. Their list was checked by the entire team (especially the pilot) and then edited from flight to flight-based on lessons learned. The list includes software, imaging, and electrical set up as well as final mechanical checks (to be sure system elements are secure for flight) and final pilot checks (to be sure flight control surfaces are reacting correctly). Not to mention, a number of ground tests were carried out prior to every flight test upon addition of any new mission capability. Some of the ground tests included battery life testing, communication link interference testing, telemetry range testing ground fail-safe testing, and ground waypoint testing.

All in all, safety was undoubtedly at the forefront of operations and design throughout the design, construction, and testing phases of system development. Using the competition rules, risk assessment, and prior experience the team established a broad, deep understanding of both the nature and importance of safety criteria both in its ability to reduce risks to property or human life and its ability to help the plane get in the air to successfully complete its mission.

Safety Risks and Mitigation Methods

The team did not have a formal risk evaluation and mitigation process this year. The overall process followed included risks observed or calculated by members informally, alerting relevant team members of said risk, evaluating methods of mitigating risk, and deciding upon mitigation method based on time and money necessary to mitigate risk. Some risks were unable to be mitigated based on the time available to be completed versus severity level. These risks were always a low threat level with a low frequency of occurrence. In these cases, if they were flight critical, the team noted such to appropriate staff at the test field and had backup systems available in case of a damaging landing.

Appendix 1: Testing Schedule

Test Type	RC/ Autopilot	Components/ Systems	Speeds Requested	Height Requested	Distance Requested	Time requested	Remarks
Ground	RC	Software, Electrical	N/A	N/A	(Subject of test)	N/A	Telemetry Links Distance Testing: autopilot to computer, autopilot to controller
Ground	N/A	Imaging	N/A	N/A	(Subject of test)	N/A	Camera Range Testing
Ground	RC	Electrical, Mechanical, Software	10% increments up to full throttle	N/A	N/A	(Until battery dies)	System battery life testing
Ground	RC	Software, Electrical, Mechanical	25 mph	N/A	N/A	N/A	Test RC fail-safe system
Ground	RC	Software, Electrical, Imaging	N/A	N/A	N/A	N/A	Interference testing
Flight 1	RC	Electrical, Software	10% increments up to full throttle (as pilot sees fit)	200-400 ft	.25 mi	20 minutes	Prove accurate software readings on the ready to fly plane
Flight 2	RC	Electrical, Mechanical	10% increments up to full throttle (as pilot sees fit)	200-400 ft	.25 mi	20 minutes	Proving successful mechanical and electrical integration with RC controller in flight
Flight 3	RC	Electrical, Mechanical, Software	10% increments up to full throttle (as pilot sees fit)	200-400 ft	.25 mi	20 minutes	Proving successful mechanical, electrical, and software integration
Flight 4	RC	Electrical, Mechanical, Software	10% increments up to full throttle (as pilot sees fit)	200-400 ft	.25 mi	(Until critical low battery)	System battery life testing in flight
Flight 5	RC	Electrical, Mechanical, Software	(Subject of test)	(Subject of test)	.25 mi	N/A	Max speed pilot is comfortable with, stall testing
Flight 6	RC	Full	10% increments up to 50%	200-400 ft	.25 mi	20 minutes	Proving successful mechanical, electrical, software, and imaging integration; determine competition speed
Ground	Autopilot	Software	N/A	N/A	.25 mi	20 minutes	Walking plane around to ensure its mechanical system correctly responds to waypoint communication
Ground	N/A	Imaging, Software,	N/A	N/A	N/A	N/A	GPS data sent to imaging system and

		Electrical					paired correctly with images to determine location where each picture was taken
Ground	Autopilot	Software, Electrical, Mechanical	25 mph	N/A	N/A	N/A	Test autonomous fail-safe system
Flight 7	Autopilot	Electrical, Mechanical, Software	(Competition speed*)	200-400 ft	.25 mi	20 minutes	Proving successful integration of mechanical, electrical, and software systems when flying autonomously
Flight 8	Autopilot	Full	(Competition speed*)	200-400 ft	.25 mi	20 minutes	Proving successful integration of full system
Flight 9	RC	Electrical, Mechanical	(Competition speed*)	200-400 ft	N/A	N/A	Reverify RC flight and general operation.
Flight 10	RC	Electrical, Mechanical, Software	(Competition speed*)	200-400 ft	N/A	N/A	Start stepping through setup documentation, attempt setup of airspeed sensor and start calibration of autopilot flight modes
Flight 11	Autopilot	Electrical, Mechanical, Software	(Competition speed*)	200-400 ft	N/A	N/A	Finish calibration of autopilot flight modes, tune autopilot to achieve better performance and better waypoint capture
Flight 12	RC	Electrical, Mechanical, Software	(Competition speed*)	N/A	N/A	N/A	Attempt verification of interoperability, cancelled after 5min of flight due to 25KTS wind gusts and heavy turbulence.

TABLE 2 – TEAM TESTING SCHEDULE