AUTONOMOUS UNMANNED GROUND-AERIAL SURVEILLANCE SYSTEM FOR AREAS WITH AN UNDERDEVELOPED SECURITY INFRASTRUSTURE

by

Steven Tyler Padgett

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Applied Energy and Electromechanical Systems

Charlotte

2018

Approved by:

____
Dr. Aidan Browne

____
Dr. Wesley Williams

____
Dr. Maciej Noras

ABSTRACT

STEVEN TYLER PADGETT. Autonomous unmanned ground-aerial surveillance system for areas with an underdeveloped security infrastructure. (Under the direction of DR. AIDAN F. BROWNE)

Events held in areas without a well-developed security infrastructure, such as the Tour de France and temporary military bases, can be difficult to secure and survey. In these situations, an unmanned aerial system (UAS) can prove useful to get surveillance in the air and around objects. These systems have become readily available and can be adapted for many uses, despite having a limited runtime. Other works have developed systems providing power through a tether system and fixed position landing platforms for recharging the UAS; however, both solutions limit mobility of the UAS. This paper details an alternative solution by combining a UAS with a mobile unmanned ground vehicle (UGV) containing a landing platform for recharging the UAS. This results in a flexible autonomous surveillance system that enhances the limited flight time of traditional UAS systems. For testing, the system was set to travel to and survey a given GPS location before returning to its initial location with fully autonomous operation. Ultimately, the system traveled to and surveyed the location as desired. Upon return, the UAS targeted and descended to the landing platform. Due to the inaccuracy of the GPS unit used in the prototype, the UAS was only able to land onto the approximately half the time. Once the UAS landed, the entire system could either return to its original location or advance to its next waypoint. Analysis of results show an IMU (inertial measurement unit) sensor and more accurate GPS location would be needed to make the system reliable.

ACKNOWLEDGMENTS

I would like to acknowledge Dr. Aidan Browne for his assistance, knowledge, and for providing most of the equipment used for this research. He has also worked with me on other research projects and given advice on numerous occasions. I would also like to acknowledge Dr. Wesley Williams and Dr. Maciej Noras for their valued input and knowledge during this process. I appreciate the financial and emotional support of my family and what they have taught me over the years. I would also like to thank God for his support and guidance along the way. I could not have accomplished this work without the people mentioned above and am grateful for all their support.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

Events held in areas without a well-developed security infrastructure can be difficult to secure and survey. In these situations, an unmanned aerial system (UAS) can prove useful to perform surveillance in the air, around objects, and under objects. These systems have become readily available, can be adapted for many uses, and are popular for video capture. Recent changes in technology have favored the use of batteries over fossil fuels to power these systems, despite having a limited runtime. A prospective solution to this limit is use of recharge stations, although, as implemented in the past they do not allow for mobile operation. A solution to this run-time constraint could be combining a UAS with an unmanned ground vehicle (UGV) for extended range and mobile charging operation.

1.1. Related Work

Past research has been conducted for UAS/UGV combined systems; some of the uses of these systems have been for disaster relief, military applications, mapping and finding obstacles, target tracking, and extending runtime. Each of these uses is summarized below.

1.1.1. Disaster Relief

A UAS-UGV combination was created by Connic Phan for the detection and fighting of wildfires (Phan and Liu). Their publication proposes a hierarchical control framework. The framework uses a mission planner to control individual system actions. This controls the coordination between individual systems based on capabilities and task allocation. The simulation framework has UAS and UGV systems communicating directly and by a central computer regardless of the design intent of the system. (Phan and Liu)

Moreover, Tamara Petrovic created an aerial-ground robotic system for general disaster response situations. The ground vehicle in this system is meant to aid the UAS's limited lifting ability and flight time and serves as a landing platform for the UAS. The UGV transports the UAS without the energy losses from hovering; the UAS provides an extra degree of freedom for the UGV with the ability to lift it over obstacles. The task for this system is to close a simulated valve after a disaster hits an industrial environment. At the time of the article, the UAS had demonstrated the ability to pick, place, assemble, and manipulate objects while the rest of the system was demonstrated by a simulation in Gazebo. The system needs a map of the environment to start, which could be done with a small UAS. The system would first need to find what possible solutions it can use to shut off the value, and then decide on what method to use. To do this, the system must define the problem, determine non-local tasks, eliminate redundant tasks, then create the schedule of events before execution. While a system like this would increase flexibility due to the UAS being able to lift the UGV, a lightweight UGV would be unable to recharge the UAS battery. (Petrovic et al.)

1.1.2. Military Applications

A report authored by William Fyfe reviews past studies and plans for UAS and UGV collaborations for military use. In the report, the effectiveness of using both UAS and UGV systems is shown to greatly benefit military operations with the uses of biological detection, payloads, target recognition, and situational awareness. Technical issues were noted to include docking the UAS to the UGV, command and control, and autonomous system navigation and integration. Although these issues were mentioned, details were not specified and the issues are to be addressed in future work. Additional plans included beyond line-of-sight control, extended ranges, dynamic deployment, and IED detection. This work highlights difficulties and improvements areas needed for viable UAS-UGV systems. (Fyfe and Johnson)

A system developed by Pavlo Rudakevych is meant to be man-portable and carry a payload of 75Kg. The system is a UGV with a parachute-like wing with a small gasoline engine to propel it, allowing for flight. The nature of this system does not allow for a stationary hover and is radio controlled, but has the benefit of a single system having the capability to drive and fly. The system had a few technical issues but did have a successful flight and could be used to scout out areas ahead of troops. This kite-like system could potentially travel extended ranges, but lacks the freedom to hover, increase altitude in the same location, or travel backward. (Rudakevych and Yamauchi)

A robotic buoy and UAS landing platform was designed by Jordan Gruber that could be used to secure coastal areas. The buoy floats on water and can be used to deploy sonar buoy’s and has a detachable UAS landing platform. The landing platform was designed to house solar panels for charging batteries although it did not contain them at the time of the article. To secure the UAS onto the platform at sea, a hinged net or magnets were discussed. For off-shore use, weight and the viscous forces when traveling through a liquid would have a larger hindrance to efficiency than a ground-based system. (Gruber and Anvar)

1.1.3. Mapping/Obstacle Detection

Jianqiang Li used a UAS to detect and map obstacles on the ground for path planning purposes. Two path planning algorithms are used as a hybrid system to optimize the global and local path using the UAS and UGV data. Preliminary tests were performed by using geometric shapes and a camera imaging. The results show the effectiveness of this approach for commonly recognizable shapes in a simulation. This shows the target recognition capabilities for the UGV-UAS system, however, the UGV and UAS in this case work as individual systems that communicate instead of a coherent system. (Li et al.)

As noted by Steven Dellenback, negative obstacle detection is difficult for a UGV because of low sensor placement. Traditionally downward sensors were used on the UGV and were raised on a mast for a higher perspective of the terrain. In their work, a UAS was used to improve this detection by allowing for a higher, leading sensor perspective. The fusion of information from the elevated perspective as well as the UGV is used to increase the effectiveness, allowing for greater travel speeds and a more detailed view of the terrain. While this system uses an UAS and UGV, the UAS is primarily used as a sensing unit for the UGV instead of the UGV aiding the runtime of the UAS. (Steven et al.)

1.1.4. Target Tracking

In the case of Carol Cheung, a UAS and UGV were developed to track a soldier. The UAS surveys an area to find a person or other operator-located target. Once the target was found, the location was shared with the UGV. The UGV would then go to and find the target. The UAS and UGV would track the target for better information or in case either system lost the target, it could re-locate the person using information from the other system. Vision and high accuracy inertial measurement units (IMUs) were needed to perform such a task. Systems like this could be used for tracking a potential target during a survey routine. (Cheung and Grocholsky)

As noted by Mark Moseley, a UAS-UGV combination was enhanced to improve line-of-site communication and vision-based tracking. Autonomy also allowed for coordination and the ability to influence each other with the intent of interaction between multiple systems. Mark Moseley's work allows for collaborative tasking of various units to search for and pursue a target. Systems like this could also be used for tracking a potential target during a UAS survey routine with a UGV aiding runtime (Moseley et al.)

1.1.5. Extending Runtime

Francesco Cocchioni devised a recharging platform and a UAS that can autonomously navigate, land, and recharge. The UAS has an onboard computer and camera for image-based flight controls that search for either of the two fiducials on the landing platform during landing. When landing, the UAS looks for a larger fiducial first but as it descends towards the platform, a smaller fiducial is then used to keep in its visible range. The smaller fiducial is similar, but has the arrow in a different location to distinguish it. As the UAS touches down, a passive centering system consisting of four cone-shaped indentions corrects for small alignment errors. On the platform, a battery charger connects to replenish the battery for later use. An advantage to this approach could be continuous operation of the recharge platform if it's located near a source of energy, whether a renewable or an electrical main. Disadvantages come from the fixed location of the platform, requiring personnel to relocate. (Cocchioni et al.)

Lida Zikou demonstrated a system to power the UAS through a tether attached to a power supply on the ground. This system achieves prolonged flight times and eliminates the onboard battery, reducing the weight of the UAS and increasing its efficiency. The tether system autonomously adjusts the tension on the tether to avoid excessive slack to avoid effects on the flight characteristics of the UAS. The tether is unwound from a rotating drum mechanism and passed through a spring; which is used for tension feedback. The tether would provide continuous operation but limited maneuverability and travel distance. (Zikou et al.)

Paulo Kemper created a few design concepts for either replacing or recharging the batteries of a UAS. One design was to land on a mat with various straight power rails to connect each cell of the battery for charging. Another had the rails laid out in a circle with a bowl shape to allow the UAS to slide into position, correcting for minor landing errors. A concept included a honeycomb landing pad that allowed for large errors in landing while being able to make a connection. Finally, a concept for landing to deposit a battery and then pick up a new battery was discussed. This system would allow continuous operation and replacing batteries versus recharging them would allow for much quicker redeployment. The complexity of this system may have been a hindrance to the real-world development, and the fixed platform location has the same disadvantages mentioned before. (Kemper et al.)

While a tether system could solve the limited runtime issue of a UAS, allowing for continuous flight, it greatly limits the range of the UAS and prevents flights over or through structures that the tether could entangle. Stationary landing platforms would allow the UAS to land and recharge when needed. However, the UAS would be limited by the locations of the platforms, having to fly from one location to another on its own and choose nearby locations to recharge. Alternatively, a floating UAS platform could serve to recharge or relay information from a UAS to improve security and response time along shorelines. Similarly, a small ground vehicle can help extend the runtime of the UAS by carrying it, but for the ground vehicle to be lifted over obstacles, it must be small and lightweight. Such a lightweight UGV is unlikely be contain a large enough energy source to recharge the UAS, or transport it long distances. A larger UGV could drive over small obstacles and contain a large enough power source to travel long distances and recharge the UAS multiple times.

An unmanned ground vehicle and an aerial system could work together to create a flexible autonomous surveillance system without the limited flight time of traditional UAS systems. This system would have other potential applications including but not limited to the security of a temporary military base and events in the open like the Tour de France. Applications could also be adapted for infrared cameras for thermal inspections, including solar and wind farms.

CHAPTER 2. SYSTEM OVERVIEW

A system containing an unmanned ground vehicle (UGV) with a landing platform and an unmanned aerial system (UAS) was developed to travel and surveil an area before returning to its starting location. Software was developed for both the UGV and UAS to allow them to make decisions and coordinate with each other autonomously. To accomplish this, the LabVIEW development environment was selected because of equipment availability and simplicity of the programming language. For the system to operate using LabVIEW, network-published shared variables were implemented for UGV-to-UAS communication. A myRIO embedded controller was used on the UGV, which was configured to create a wireless network access point; a second myRIO used on the UAS was configured to connect to the UGV’s access point with a static IP address. A laptop was used for the controls and parameter adjustment and was also set to connect to the UGV’s access point. The network connections are shown in Figure 1.

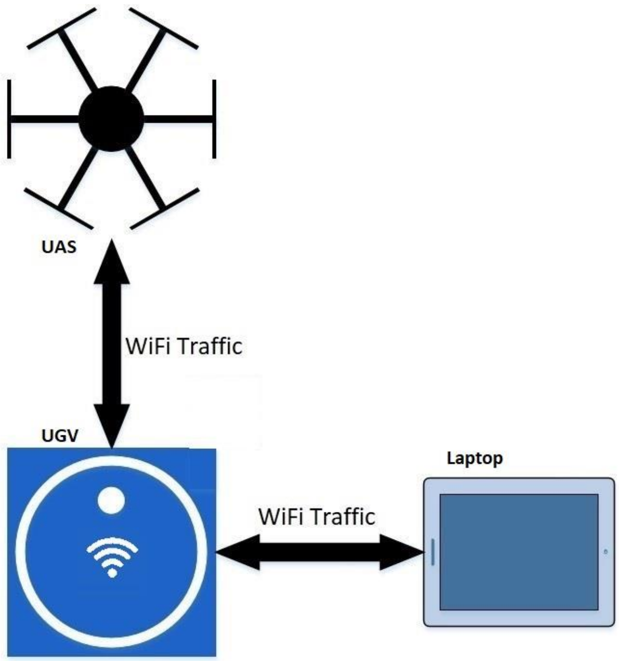

Figure 1: The network configuration between the UGV, UAS, and computer

The myRIOs on the UGV and the UAS have their own set of shared variables for their controls and adjustment of parameters. To allow for human operation of the system, a control program was created that ran on the laptop. This program took all the user assigned parameters and commands and wrote them to shared variables on the UAS and UGV. Independent loops were added to the UGV and UAS for reading the shared variables and assigning them to local variables on each controller. The local variables were then used throughout the programs to address the access, read, and write delays associated with publishing shared variables on the wireless network. This architecture allowed the UGV and UAS to communicate with each other and the operator. For safety, each myRIO would ping the operator's laptop twice per second to ensure it had a connection; if the connection to the operator was lost, the UAS would return to base to land and the UGV would stop operation.

CHAPTER 3. UNMANNED GROUND VEHICLE (UGV)

The main task of the UGV is to carry the UAS between destinations to conserve power and carry an energy source capable of recharging the UAS while traveling between flights. It was designed such that the UGV must be able to navigate by itself, avoid obstacles in its path, and release the UAS at a given location. The UGV was devised to be able to travel over broken surfaces with a maximum speed of around 11-12 feet per second. The UGV measures 30 inches wide and long and can handle a payload of around 50 pounds. For sensors, it utilized LIDAR, GPS, and a compass.

3.1. Hardware

The UGV is comprised of a main chassis and a landing platform. The main chassis contains the essential parts to operate the UGV while the landing platform primarily holds the UAS in position. The UGV can be seen in Figure 2.

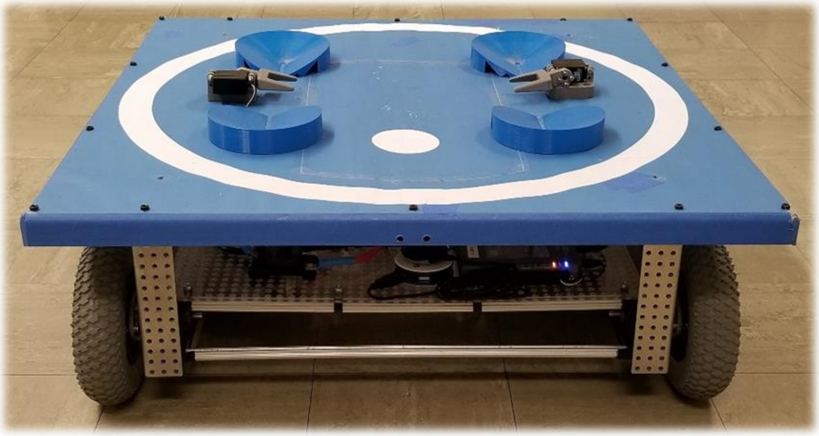

Figure 2: The UGV chassis and landing platform

The chassis has 6, 8-inch diameter, pneumatic wheels with the middle wheel being slightly lower than the others to allow easier skid-steer operation. A gearbox internal to the chassis connects all 3 wheels on each side using 4 Chiaphua Components Limited Industrial Motor (CIM) motors and Victor SP motor drivers by VEX Robotics.

Angle brackets were used to connect a 30-inch by 30-inch square landing platform frame above the chassis to each of the corners. A perforated polycarbonate sheet was mounted on top of the chassis as well as the landing platform to provide a surface that allows mounting of the electronics and a 12V, 18Ah SLA battery. The battery is connected through a resettable breaker before distributed through a fuse box to each motor and the electronics. 5V regulators are also used to power a LIDAR sensor and UAS latching servos. The main chassis and layout of the UGV are shown in Figure 3:

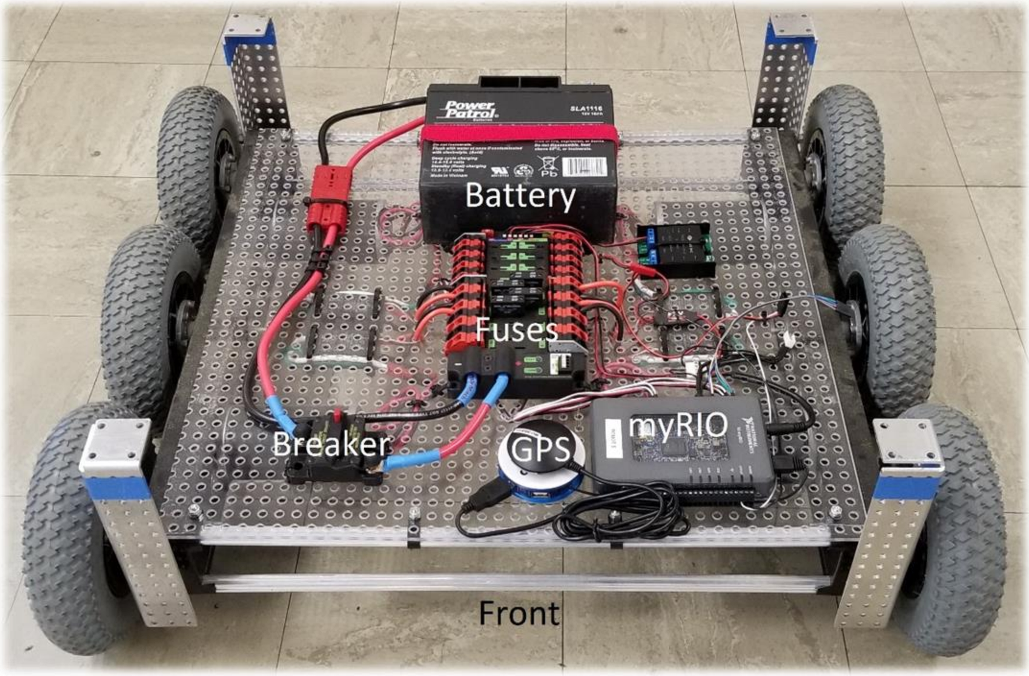

Figure 3: Main chassis and layout of the UGV

The UGV uses a SLAMTEC RPLIDAR-A2 LIDAR sensor, a GlobalSat BU-353-S4 GPS module, and a Digilent PMOD CMPS compass for sensors with a National Instruments myRIO for the controller. The LIDAR sensor was mounted on the underside, in the center of the landing platform to give it a good view of its surroundings and oriented so the angle zero was aligned with the front of the UGV. The underside of the landing platform and the placement of the LIDAR and compass are shown in Figure 4:

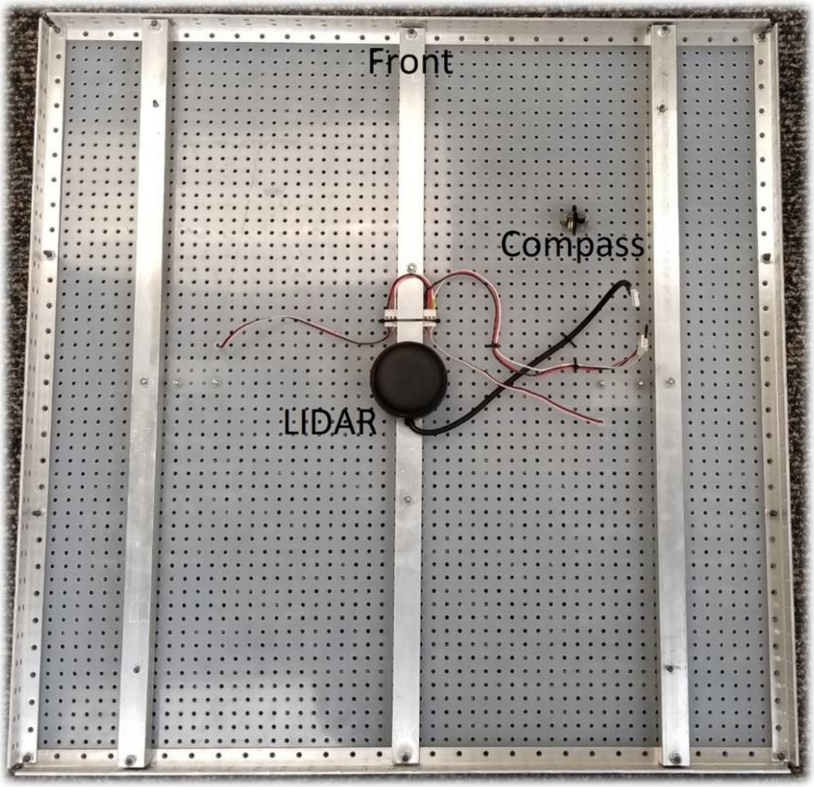

Figure 4: Placement of the LIDAR and compass underneath the landing platform

The GPS module was mounted on the main chassis toward the front center of the robot to avoid electromagnetic interference from the motors and to give it a better view of satellites. By monitoring the field strength along each axis of the compass in different locations inside the chassis, it was determined that the compass should be mounted on the bottom of the landing platform, away from all other sensors, motors, and controllers.

For the UAS to recognize where the UGV was, a blue fiducial containing 2 white circles was created to stand out from the surrounding environment. The circles were positioned so they would be underneath the webcam for close range use, and around the landing guides for use at farther distances. The fiducial was drawn, plotted, and mounted onto the landing platform. The resulting top of the landing platform is shown in Figure 5.

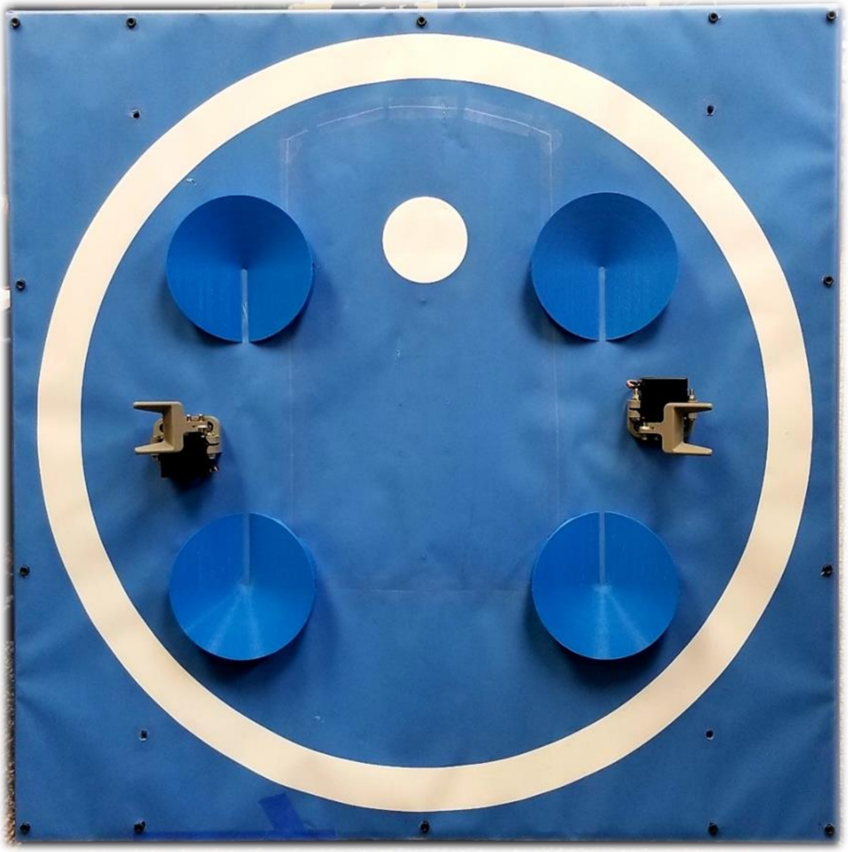

Figure 5: Fiducial on the UGV

With factors such as wind, sensor errors, and control imperfections, the UAS is unlikely to land in the same position each time. On the other hand, the UAS needs to rest in a location accurate enough to engage a charging system and for the servo attachment to interface with the landing gear. To solve this issue, cone-shaped guides were designed and 3D printed so that gravity would slide the UAS into position. The guides are shown in Figure 6.

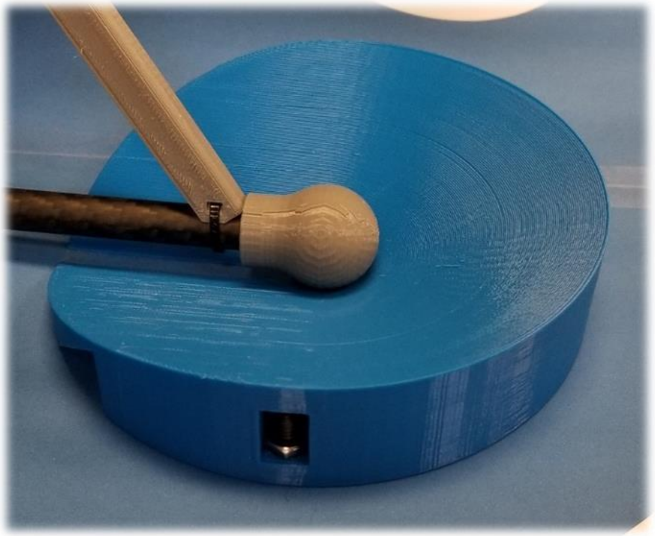

Figure 6: A guide and foot placed at each corner to align the UAS during landing

For the UAS to travel on top of the UGV across varying terrain, a mechanism was developed to hold the UAS to the landing platform. For this, two servo mounts (one on each side) were created and 3D printed to mount servos to the landing platform. They were positioned at the height needed to reach above the bottom of the landing gear. A servo attachment was also designed and printed to reach around the landing gear, preventing it from sliding or coming loose from the platform. Due to a ledge on the landing gear, the inner faces of the attachment were made at an angle to prevent any binding on the ledge. When it is time for the UAS to take off, the servo rotates, releasing the UAS. Once the UAS lands, they rotate back to secure it. The servo mechanism holding and releasing the UAS are shown in Figure 7.

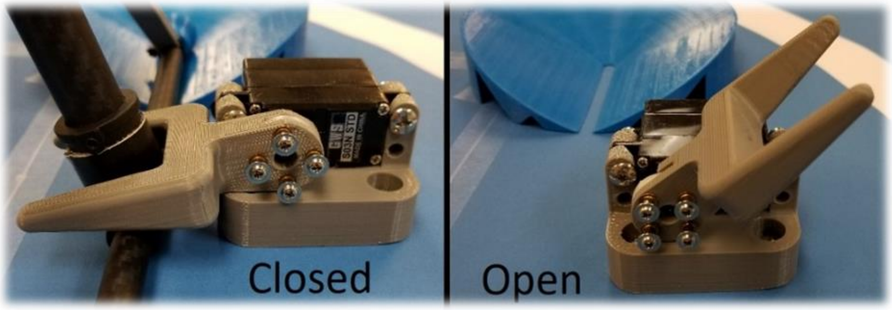

Figure 7: Servos holding and releasing the UAS

Due to the T shape of the landing gear on the UAS, the servo attachments were designed in fork shape to go around and over the landing gear. The connector at the T joint also has a ledge that the servo attachment could get caught on. For this, the inside of the fork shape has a chamfer angle to ensure the edge to the T joint gets pushed into the fork. In addition, the plastic spline attachment provided with the servos was thin, so the 3D printed attachment mounted around both sides to help prevent any bending and cracking of the spline attachment.

3.2. Sensor Data Processing

The UGV makes use of the LIDAR, GPS, and compass for navigation and obstacle avoidance. Sensors collect data that is processed and stored independently for use by control algorithms to calculate the desired outputs. The LIDAR has an 18-meter range, 360-degree view, and around 10 Hz rotation speed. It used serial communication to stream data from surrounding objects. The communication port had to be configured before sending the LIDAR a reset command. The read buffer was then flushed, and the motor enable pin was set high. 2 seconds afterward, the main control algorithm starts and the LIDAR receives a scan command. Before angle and distance data was received, the LIDAR sent a 7-byte header. To find the header, a loop was created to read each byte and compare it to the previous byte. Once the correct sequence of characters was found, the loop ended. The next loop read the data when available, in packets of 5 bytes, and pushed them to a queue. This was necessary to allow the processor the read the data at a high enough rate to prevent a buffer overflow. A separate loop dequeues the data and arranges it into pairs of angles and distances. The check bits are used to verify valid data before the bits are reordered and scaled to convert the angle data into degrees and the distance data into meters before they are assigned into an array. Distance measurements below 0.38 meters and their respective angle measurements are removed from the array since these are either invalid or reflecting off a component within the chassis of the UGV. The data is then sorted into sections to the front, left, rear, and right sides of the UGV. The minimum value for each section is recorded for the motor control algorithm to use. The sections around the UGV are shown in Figure 8.

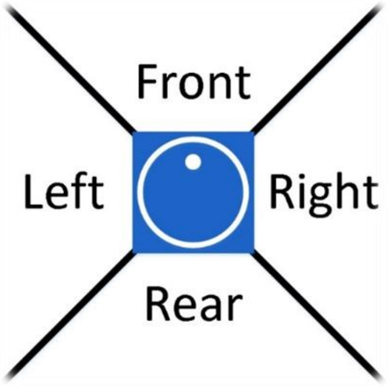

Figure 8: The LIDAR data sections

The data is processed for obstacles in front of the UGV. It was decided that the UGV shouldn’t react until an object is within 2 meters along its travel direction and within a half-meter clearance from the sides of the UGV. With this, equations were developed to determine if the object is within the defended area. To change functionality based on an obstacle, a flag is triggered if any of the datasets contain values that satisfy both equations below:

$$|L \sin \theta| < \frac{W}{2} + 0.5 \tag{1}$$

$$|L \sin \theta| < \frac{W}{2} + 0.5 \tag{2}$$

L is the distance from the edge of the chassis to the scanned point, θ is the angle of the scanned point, W is the length and width of the chassis. The 0.5-meter offset is to ensure the UGV clears the object by half a meter. The highest and lowest angle measurements that satisfy the equations are recorded for the main control algorithm to use. Once the UGV ends its program, the motor enable pin is set low and the port is closed.

The GPS sensor used has an update rate of 1 Hz and a tracking sensitivity of -163 dBm. Serial communication is used through the USB port of the myRIO using standard NMEA sentences of data. The myRIO was programmed to read the bytes from the GPS and concatenate them until the end of a sentence is read. If the sentence matches the GPGGA (GPS fix information) or GPRMC (recommended minimum data for GPS) dataset, the appropriate case statement is selected. For the GPRMC dataset, the GPS status (valid or not), speed (in knots), and course (in degrees from north) are used. The speed is multiplied by 0.5144 to convert into meters per second. In the GPGGA dataset, the UTC time, longitude (degrees), latitude (degrees), directions (north or south and east or west), number of found satellites, and altitude (m) are used. All the values are read as strings in LabVIEW, then the speed, course, latitude, longitude, and altitude are converted from a string a double precision number to allow for the appropriate calculations. The GPS coordinates were represented by positive position values and north or south and east or west hemispheres. GPS coordinates could also be represented by positive and negative position values, their fore the latitude was negated if the hemisphere was south and the longitude was negated if the hemisphere was west. This created positioning represented by positive and negative coordinates instead of being represented by a position and hemisphere.

The compass used is a 3-axis digital compass with 2 milli-gauss field resolution in ±8 gauss fields. It uses I2C communication and provides the intensity of the magnetic field. To calibrate the compass, the operator triggers the calibration algorithm and rotates or drives the UGV though a full rotation. Meanwhile, the calibration algorithm records maximum and minimum magnetic field strengths to compensate for variations in the magnetic field and sensor. Once the operator turns the calibration routine off, it calculates the average and range of each axis to center the values on zero and normalize them to plus or minus one. It then uses the adjusted values and inverse tangent function to calculate its orientation using north as 0 degrees, and east as 90 degrees. The previous 5 values are averaged before recording the orientation. The previous 2 values are also compared for equality and if they match, a flag is triggered indicating the data is invalid.

3.3. Motor Controls

The controls for the UGV contain 2 algorithms; one to calculate and drive motor controls, and the other to make decisions. For the motor controls, maximum and minimum throttle percentages are calculated 50 times per second based on the distance from obstacles in front, to the sides, and to the rear of the UGV. To ensure smooth operation, the forward throttle is not allowed to change by more than 2% while the rotation throttle is not allowed to change more than 1% for each in each iteration, resulting in ramping throttle values. At any point in the routine, the operator can disable the motors, making the forward and rotation throttle set immediately to zero instead of ramping down. The algorithm also disables the motors in the same fashion if the UGV controller loses connection with the operator or if the program receives an end request.

Based on the dimensions of the UGV, if any object is closer than 0.55 meters from the sensor, the rotation throttle will be reduced to zero, regardless of commands, to ensure the corners of the UGV do not hit an object. To determine the speed the UGV can travel towards an object, an exponential curve was determined to perform as desired. With testing by driving the UGV towards obstacles, the multiplier and offset were determined to create the desired performance. From this, the forward throttle (*T*), the maximum value is determined by the equation below:

$$T = 4(D - 0.4)^2 + 8 \tag{3}$$

D is the distance from the LIDAR sensor to an obstacle in front of the UGV, 0.4 is the distance from the LIDAR to the edges of the UGV, and 8 is the minimum throttle required to make the UGV travel. If the LIDAR does not see a reflection, the resulting *D* value will be equal to zero and the system will change the maximum forward throttle (*T*) to 100%. For the UGV to travel backward, the forward throttle is negative and used the same equation, except using the distance to an obstacle behind the UGV. At any time, if the LIDAR sees an object within 0.4 meters in front, the maximum throttle value will become zero, likewise an object within 0.4 meters behind will make the minimum become zero. This operation prevents the UGV from hitting objects while allowing it to move away from them. The resulting throttle values are then passed into a motor mixing program in the same loop to create the needed pulse widths to drive the motors.

The motor drivers operate at a frequency of 50 Hz, with a pulse with of 2 ms to making the UGV drive at its maximum speed forward, and a pulse with of 1 ms resulting in its maximum speed traveling backward, varying linearly in between. The throttle percentages are first coerced to fit within the maximum values set by the operator. Since a positive rotation throttle rotates the UGV clockwise, it gets subtracted from the forward throttle for the left side of the UGV and added to the right side. Each side is divided by 4000 and coerced between 0.05 and 0.1 to create the desired duty cycles.

3.4. Autonomous Driving

The main control algorithm for the UGV contains six primary functions; 1) *manual*, 2) *manual with obstacle avoidance*, 3) *sit*, 4) *travel to*, 5) *release UAS*, and 6) *return*. If the operator makes a function change, the algorithm will exit its current function and continue to the selection function. If at any time the UGV disconnects from the control computer, it will default to the *sit* function, otherwise, it will go to the function selected by the operator. The *manual* function takes the inputs from the operator and sends them directly to the motor control algorithm. The *manual with obstacle avoidance* function performs the same as the *manual* function until an object is in the path of the UGV. Once an object has been detected, it compares the absolute values of the maximum and minimum angles that satisfy the requirements of an object in the path of the UGV. The angle with the smaller absolute value indicates the direction the UGV will rotate while an equation was developed to give the desired rotation performance for avoiding obstacles. With this, the rotation throttle (*R*) was determined by the following equation:

$$R = \pm\, 1.5T \tan^{-1} \theta_o \tag{4}$$

T is the forward throttle and θ_o is the corresponding angle to the object (whichever has a smaller absolute value). In addition, if an object appears too close to the sides of the

UGV, the rotation throttle is restricted to prevent any rotation in the direction of the object. The resulting values are sent to the motor control algorithm. For example, Figure 9 shows the obstacle detection areas of the UGV.

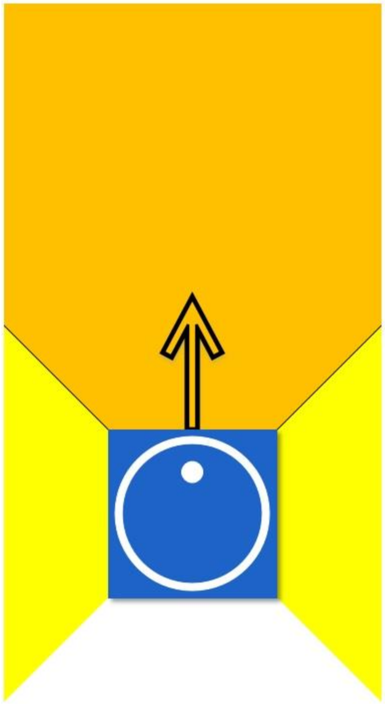

Figure 9: Obstacle sections around UGV

The orange section in front of the UGV shows the obstacle detection area. If an object appears in this section, the angles will be compared to determine the direction to steer away from the obstacle. If the obstacle leaves the orange section and enters the yellow section, the rotation throttle will not be allowed to rotate toward the side of the obstacle. Once the obstacle exits both sections, the UGV will return to the selected navigation method. This obstacle avoidance method is also used if an obstacle is detected in the *travel to* and *return* functions.

When the UGV is in one of the autonomous functions, it will cycle through them without entering either *manual* mode. Each function will end either when certain criteria are met, the operator tells it to go to the next function, or the function is changed. In the *sit* function, it assigns a forward and rotation throttle of zero to prevent it from moving. From the *sit* function, if the operator tells the system to go, it will change to the *travel to* function and save its current location. When traveling to or returning, the distance from the setpoint is calculated using the haversine formula (Brummelen) shown below:

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi_1 * \cos\varphi_2 * \sin^2(\Delta\lambda/2) \tag{5}$$

$$c = 2 * \mathrm{atan}^2\left(\frac{\sqrt{a}}{\sqrt{1-a}}\right) \tag{6}$$

$$d = R * c \tag{7}$$

Where φ_1 and φ_2 are the latitude position and setpoint, and λ_1 and λ_2 are the longitude position and setpoint. The mean radius of the Earth, R, is taken to be 6,371 kilometers. To simplify the calculation, the position of the test field was then used to approximate a linear relationship between latitude, longitude, and distance. The approximation took each degree of latitude to be 111.195 km and each degree of longitude to be 90.745 km. The resulting latitude error (La_e) and longitude error (Lo_e), in meters, was calculated by the equation below:

$$La_e = 111195(La_s - La) \tag{8}$$

$$Lo_e = 90745(Lo_s - Lo) \tag{9}$$

La_s is the latitude setpoint, Lo_s is the longitude setpoint, La is the current latitude, and Lo is the current longitude. The error distance (D_e) from point to point is then calculated by the following equation:

$$D_e = \sqrt{{La_e}^2 + {Lo_e}^2} \tag{10}$$

The forward throttle is then assigned by multiplying the error distance by 5. The angle correction (θ_c) is calculated based on the needed angle and current angle of the UGV (θ_v) following equation:

$$\theta_c = \tan^{-1}\frac{Lo_e}{La_e} - \theta_v \tag{11}$$

The rotation throttle is then assigned by dividing the angle correction by 3, but it cannot exceed 3 times the square root of the throttle value. This allows rapid turns without overshoot at high throttle values. If an object is detected, the rotation is then determined by the obstacle avoidance function formerly discussed. Once the UGV gets within 2 meters of the set location, it advances to the *release UAS* function.

The *release UAS* function assigns forward and rotation throttle values of zero and waits for 1 second to ensure the UGV has come to a halt. The UAS is released by updating a PWM signal to the pair of servos and updates a shared variable on the UGV named fly. The UAS reads this variable, takes off and performs its functions, and lands back onto the UGV. After landing, it resets the shared variable, read by the UGV. The UGV then updates the PWM to the servos to secure the UAS before advancing to the *travel back* function. The *travel back* function uses the same equations as the travel to function, but the setpoint is set to the UGVs original location from when it started to travel. Once done, it returns to the *sit* mode.

CHAPTER 4. UNMANNED AIRCRAFT SYSTEM (UAS)

The main task of the unmanned aerial system (UAS) is to record video while it flies around the desired location and land back onto the ground robot to be carried between locations. For this prototype, it was determined that the UAS will perform one rotation at the desired location. The UAS consisted of a carbon fiber frame with 6-rotors and an opposing rotor-to-rotor distance of 0.695 meters. The UAS uses 14.8V lithium batteries and can handle a payload of around 1.8 kilograms. It makes use of a LIDAR, GPS, and Compass for navigation and a proprietary flight controller for stability.

4.1. Hardware

The structure of the UAS is a Tarot 680 6-rotor carbon fiber frame with Hobbywing Platinum Pro 30A motor controllers, Tarot 4006 brushless motors, and 1355 carbon fiber propellers. Two 14.8V lithium-polymer batteries ranging between 5Ah and 7Ah were run in parallel to satisfy the power requirements of the UAS. The power from the batteries is regulated to 12V for the National Instruments myRIO controller. The UAS makes use of a DJI A2 flight control system, Garmin LIDAR-Lite V3, GlobalSat BU-353S4 GPS module, Digilent Pmod CMPS compass, and a National Instruments myRIO for the controller. The UAS can be seen in Figure 10.

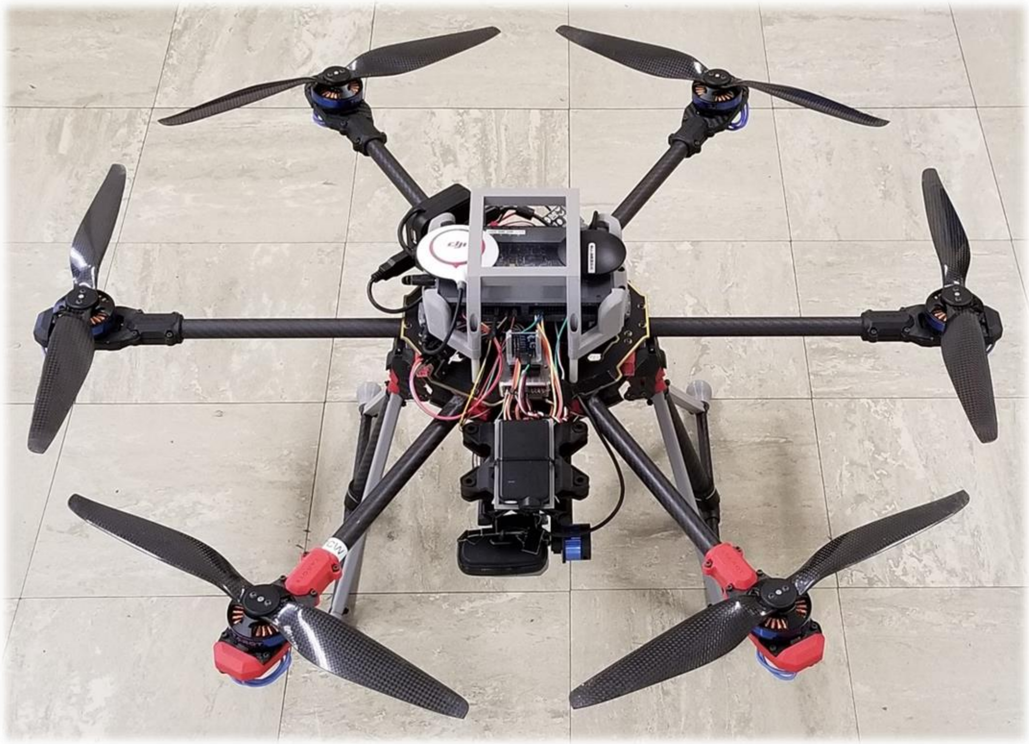

Figure 10: The Unmanned Arial System (UAS)

The DJI A2 system contains a MEMS (microelectromechanical system) sensor, a PMU (phasor measurement unit), a GPS/compass unit, interface/indicator unit, and the A2 flight controller. The MEMS sensor was mounted on top of the bottom frame plate, in a void in the center of the top frame plate for better sensor data. The PMU was mounted to the side of the MEMS on the top frame plate, and the GPS/compass was mounted on top of the myRIO to give a clear view of the satellites. The interface/indicator unit was mounted by a bracket was designed and 3D printed to hold it onto carbon tubes that run underneath the frame to give visibility of the indicator during flight. The flight controller was mounted front-center of the top frame plate with a PWM to S-BUS converter on top to allow the myRIO to send PWM commands to the flight controller. The DJI system and S-Bus converter can be seen in Figure 11.

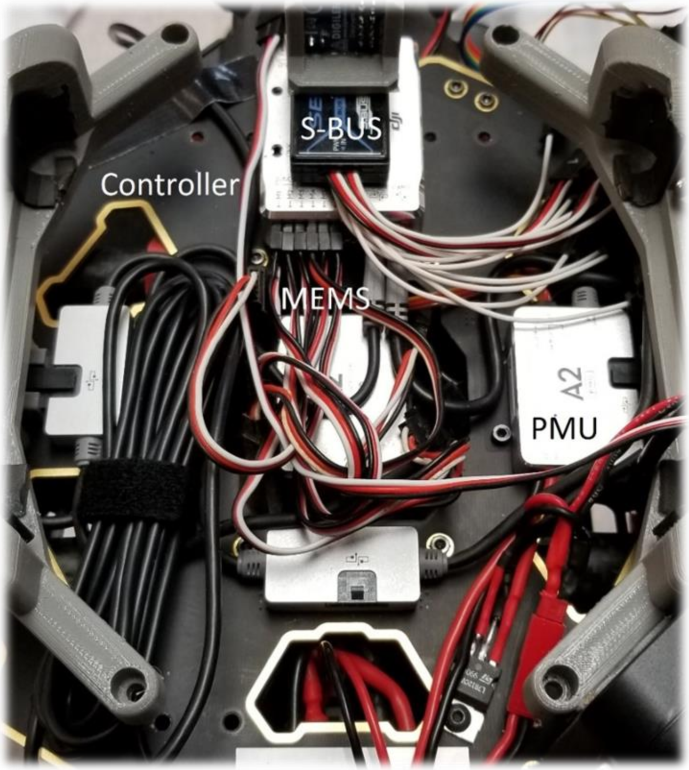

Figure 11: Mounting of the DJI A2 system (silver) and S-BUS converter (black)

The LIDAR-Lite was mounted by a bracket that was designed and 3D printed to hold it onto carbon tubes that run underneath the frame. It is important to note that most 3D printed parts were made of PLA plastic with an infill of 25%; the exception being the part holding the myRIO. The printed part positions the sensor underneath the UAS frame, as close to the center as possible while maintaining the highest possible distance to the ground to reduce any potential range problems. Another bracket that was designed and 3D printed to hold the LIDAR-Lite beside the DJI interface module. Both parts are mounted beside the batteries that were positioned to place the center of mass as close to the center of the UAS as possible. The mounting of the LIDAR-Lite, DJI interface, and battery unit can be seen in Figure 12.

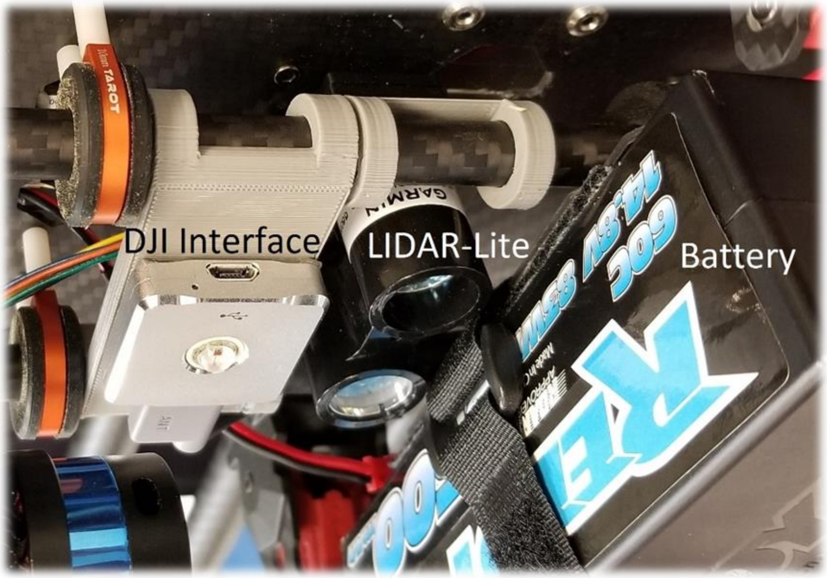

Figure 12: Mounting of the DJI interface, LIDAR-Lite, and battery underneath the UAS

Due to the size of the myRIO controller and amount of wires on the top frame plate, it was decided that the myRIO should be held above the wires and DJI system. 3D printed mounts were designed to do so, with the infill set to 10% to reduce weight and material usage over a relatively large part. To reduce high-frequency vibrations, foam pads were later added between the mounts and myRIO. The GlobalSat BU-353S4 and DJI GPS modules were mounted on top of the myRIO, at opposite ends to prevent interference in the DJI compass. The Pmod compass was mounted above the S-BUS converter by creating a friction-fit 3D printed part. This positioned it in a moderately open area in front of the GPS units and gimbal. In a case when a user installs propeller blades in the incorrect orientation, or if the system experiences a failure, the UAS may flip over. To protect the myRIO, GPS units and provide extra frame durability, a roll-over cage was designed to take any above or upside-down impacts. The mounting of the myRIO, GPS units, compass, and roll-over cage can be seen in Figure 13.

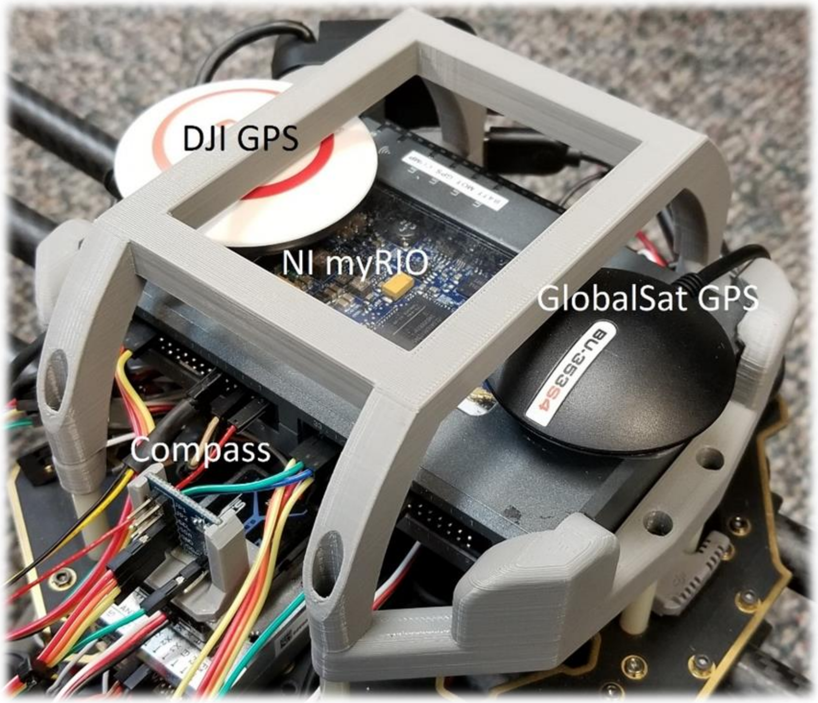

Figure 13: Mounting of the GPS and compass units

In the case of a system or sensor failure, the UAS could rapidly descend and impact the ground. The stock carbon tubes used for the landing gear were unsupported and failed to withstand such impact. As a solution, braces were designed to direct most of the force from the landing gear to the frame. This prevented low-height failures from damaging the landing gear. The landing feet from Tarot were made of rubber, so they were replaced by round feet that were designed and printed as well to reduce friction. The light gray landing gear braces and feet can be seen in Figure 14.

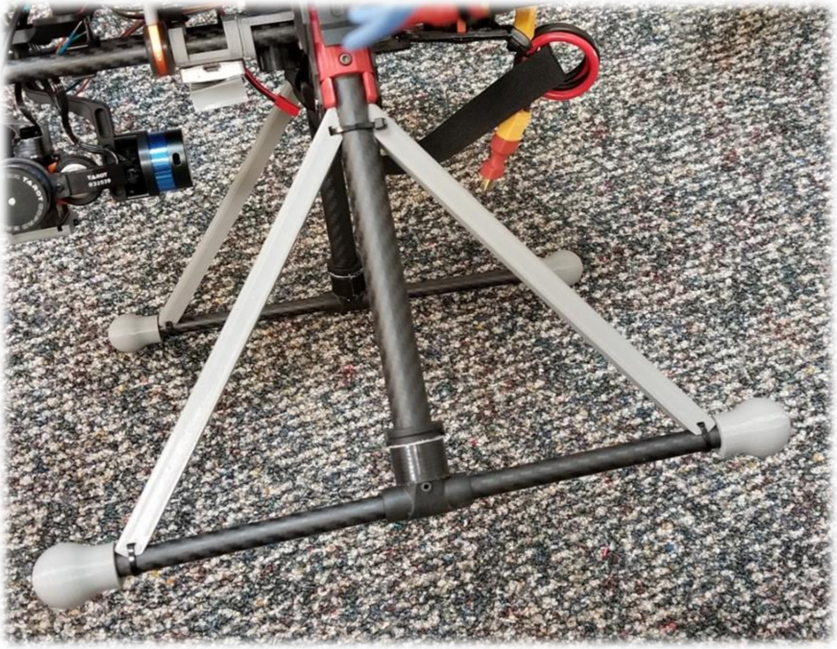

Figure 14: Landing gear braces

To eliminate the pitch and roll dynamics of the UAS, a Tarot T-2D gimbal and Logitech C525 Webcam were added. To connect the webcam to the myRIO over USB, a USB hub was added to the UAS. The gimbal is normally configured to mount completely underneath the frame. For a better field of view, the webcam needed to be as far from the ground as possible. To accomplish this, the gimbal was reconfigured to mount above the carbon tubes while a 3D printed part was used to hold the webcam in the position where a GoPro camera was intended to fit. Due its size and weight, a part was 3D printed to mount a Mobius 2 camera on top of the gimbal for video recording and transmission. The mounting of the gimbal, webcam, and Mobius 2 can be seen in Figure 15.

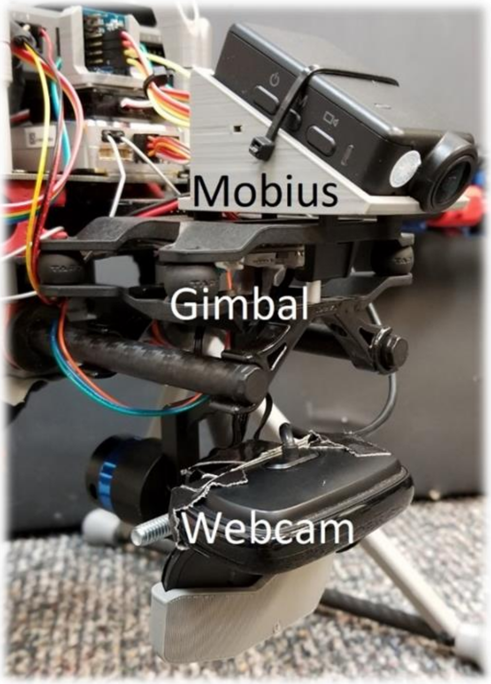

Figure 15: Mounting of webcam, gimbal, and Mobius camera

4.2. Sensor Data Processing

The myRIO was programmed to take sensor data, makes decisions, and send the necessary commands to the DJI A2 flight control system. The DJI A2 system is used to take commands from the myRIO and its own sensor data to drive the motors accordingly while making corrections to stabilize the UAS. The A2 is a fully independent system, taking external inputs from the myRIO but does not have any outputs for data. It contains a MEMS sensor for IMU (inertial measurement unit) data, a PMU for power measurements, GPS, and compass for failsafe protection. The proprietary system has a hover mode with a vertical accuracy within 0.5m and horizontal accuracy within 1.5m. Setup of the A2 system included specifying a 6-rotor frame, calibration of the MEMS and Compass, and placement details of the MEMS and GPS units. The A2 controller was also set to create a virtual boundary to prevent a runaway and to automatically land the UAS in case the battery voltage became dangerously low during testing. The compass on the

A2 could be recalibrated by a command from the user to the myRIO. This was accomplished by having the myRIO toggling the flight control between *manual* and *GPS Attitude* mode every tenth of a second, for one second. Otherwise, the A2 is held in *GPS Attitude* mode.

The GPS and compass are the same as used on the UGV and are read in the same manner. However, the dynamics of the UAS allow it to keep traveling unless it banks away from the direction of travel. For this, the UAS makes use of a derivative multiplier to control overshoot. Therefore the latitude (La) and longitude (Lo) are the latitude and longitude coordinates which are calculated by the following equations:

$$La = La_0 - D_g(La_{-1} - La_0) \tag{12}$$

$$Lo = Lo_0 - D_g(Lo_{-1} - Lo_0) \tag{13}$$

Where D_g is the derivative coefficient, based on the current and previous latitude (La_0 and La_{-1}) and longitude (Lo_0 and Lo_{-1}). The current and future coordinates are saved for the control algorithm to use.

The lithium-polymer battery used on the UAS must also be monitored for voltage for longevity of the battery as well as consistent flight control. The myRIO controller was programmed to monitor the battery voltage, allowing it to make decisions on when it needs to return and land. To allow the myRIO to read the battery voltage, a voltage divider circuit was designed with the voltage across a resistor fed to an analog input. A simple algorithm was created to read the voltage and average the last two values.

The 3-axis accelerometer built into the myRIO were used to detect how level the UAS was before takeoff so it will not take off on an incline. Since gravity on Earth can vary by location, the steady-state value of the sensor was approximately 0.99g. With this, the Z-axis acceleration was determined by subtracting 0.99g and averaging the last 10 values and sent to the control algorithm. The accelerometers were initially used to detect impacts for disabling the motors; however, due to the vibrations during flight creating excessive noise, the function was removed.

The Garmin LIDAR-Lite V3 has a 40-meter range with a +/-2.5cm accuracy and up to 500Hz sample rate. It was used in PWM mode, meaning the time the output pulse remained “high” represents the distance measured. Due to the short pulse length being too short for the processor-based inputs, the myRIO needed to use the FPGA to read the length of the pulse. For this, an algorithm was written in the FPGA to wait until the pulse went high, then record the absolute time in µs. Next, it waited until the pulse went low, and recorded the absolute time again. The difference in absolute time was pushed into an array of the last 20 read values to allow for later averaging. To fit this program onto the FPGA, the stock logic for encoders was removed. Inside the main program, another algorithm read the FPGA array and calculated the height (h) in meters of the UAS from the following equation:

$$h = \frac{\frac{\sum t}{20}}{1000} - 0.15 \tag{14}$$

Each of the 20-time measurements (t) are summed and averaged before being converted into meters and offset by 0.15 meters due to the characteristics and placement of the sensor. In case the LIDAR-Lite is out of range or experiences a failure, the current altitude minus the altitude from its takeoff position is used instead. It is also worth noting that the LIDAR-Lite’s output did not have a linear correlation to distance. This was discovered by plotting the output versus time while slowing lowering the UAS towards the ground 5 times. The resulting distance measurements are shown in Figure 16.

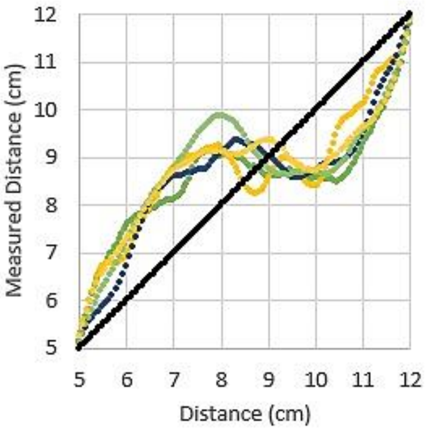

Figure 16: The LIDAR-Lite discontinuity

The colored values represent sensor data, compared to the theoretical values shown in black. Between around 8 and 10 centimeters, the internal processing seems to change modes. Due to the continuous outputs and non-linearity, the LIDAR-Lite cannot be used accurately in this range. This contributes to the reasoning for mounting the LIDAR-Lite as high as possible on the UAS

For ease of use and testing purposes, the 4 built-in LED’s on the myRIO were programmed to indicate the status of the UAS. If the compass values were within range and constantly changing (due to noise or movement), LED1 (COMP) would become solid. If the GPS had a position fix, but less than 6 satellites, LED2 (GPS) would blink every half a second. LED2 would become solid if more than 6 satellites were detected, or off if it could not get a fix. Similarly, LED3 (MOT) would blink if test throttle was activated, turn off if the motors were disabled, or become solid if the motors were enabled. LED4 (BATT) was used to indicate battery level. It was programmed to be solid if the voltage was over 15.6V, blink between 14.8V and 15.6V, or turn off if it was below 14.8V. The LED indicators can be seen in Figure 17.

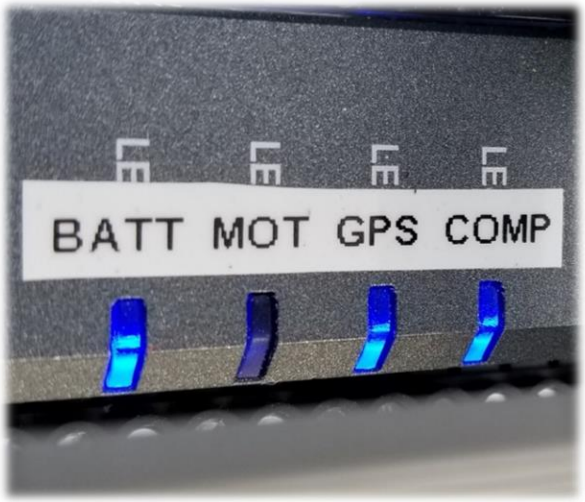

Figure 17: LED indicators on the myRIO controller

4.3. Motor Controls

Like the UGV, the UAS contains an algorithm that determines the needed commands for autonomy, and one to calculate and write the commands to the DJI A2 flight controller. The flight controller was set up for 5 PWM input channels; containing the flight mode, throttle, yaw, pitch, and roll. The flight mode was constantly held in GPS Attitude mode since the other two modes (Attitude and Manual) required a rate of change instead of an angle for pitch and roll. For simplicity, the throttle, yaw, pitch, and roll are calculated in percentages, with throttle values ranging from 0 - 100% and the rest ranging from -100 to +100%. The positive directionality for these controls is shown in Figure 18.

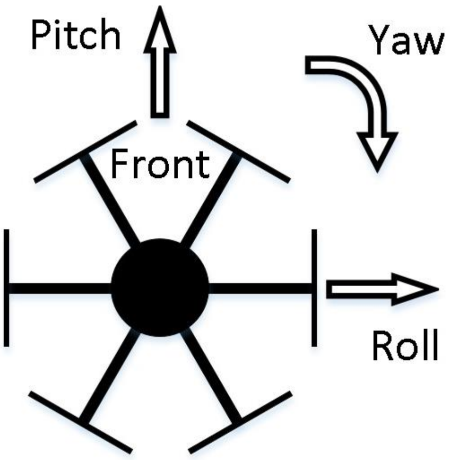

Figure 18: Positive directionality of pitch, roll, and yaw

A flag for testing, motor disable, and setting the maximum angle also took priority over the commanded values. If the test flag was enabled, the throttle was divided by two to ensure the UAS would react to changes but could not leave the ground. If the motors were disabled, a throttle value of 0% became the output. For the maximum angle, the pitch and roll were coerced to have an absolute value equal to or less than the angle percentage. In addition, the throttle was made so it could not go below 40% if the UAS was more than 0.25 meters from the ground and the motors were enabled, regardless of control state, to prevent the motors from being able to shut off mid-flight.

The motor control algorithm runs at 25Hz; under normal operation, each value can only increase or decrease by one each iteration for smoother operation. This is not true when the motors are disabled or when arming the A2 flight controller. To arm the flight controller, each time the throttle command increases from 0% the throttle gets held at 0% while the yaw, pitch, and roll are held at 100% for 0.75 seconds. To create the PWM outputs needed for the A2 and maintain the ability to fly manually from a typical radio, the A2 was calibrated to a Futaba 14SG radio system and the output duty cycles from the myRIO were experimentally adjusted to match the radio. To convert the throttle percentages ($T_{\%}$) into the needed PWM throttle duty cycle (T_D) at 50Hz, the following equation was used:

$$T_D = \frac{T_{\%}}{2380} + 0.097 \tag{15}$$

Likewise, the yaw, pitch, and roll duty cycles were experimentally adjusted to match the radio. With this, the yaw, pitch, and roll percentages ($P_{\%}$) were converted into the needed PWM duty cycles (P_D) by the following equation was used:

$$P_D = \frac{P_{\%}}{4760} + 0.076 \tag{16}$$

4.4. Autonomous Flight

The main control algorithm for the UAS contains 7 primary functions; 1) *manual*, 2) *sit*, 3) *takeoff*, 4) *traveling to*, 5) *survey*, 6) *travel back*, and 7) *land*. The function is usually selected autonomously, although the operator can skip to the next function if desired. In addition, if at any time the UAS loses GPS signal, it will advance to the *land* function and attempt to land.

In *manual*, the user selected values get passed directly into the motor control algorithm. If the connection between the UAS and operator is lost, then all outputs are set to 0%. Likewise, the *sit* function assigns all outputs to 0%. If either function is activated with the UAS in the air, the motor control algorithm will prevent a rapid descent.

When the UAS receives a fly command, it changes to the *takeoff* function and checks to make sure it is ok to take off. If the motors are enabled and the voltage is below 15.6V, the GPS does not have a lock on its location, Z-axis acceleration is not between -0.02 and 0.02 g, or the height is not in-between -0.02 and 0.125 from the ground, the UAS will not take off. If the requirements are met, it saves its current latitude, longitude, altitude, and orientation. The throttle is then set to 60% without any other outputs until the UAS reaches 0.5 meters above the ground. The loop then runs at 50Hz while correcting to a height setpoint as well as the orientation from takeoff. The height setpoint (*H_{set}*) is increased by the rise rate (m/s) divided by 20 each iteration. The equation governing the new throttle value (*T_0*) is shown below:

$$T_0 = H_{set} - (D_h(H_0 - H_{-1}) + H_0) + T_{-1} \tag{17}$$

Where *D_h* is a derivative gain, *H_0* is the current height, *H_{-1}* is the previous height, and *T_{-1}* is the previous throttle. The yaw of the UAS is used to maintain the same orientation as when it took off (*θ_t*). The equation determining the yaw (*Y*) percentage is shown below:

$$Y = 3(\theta_c - \theta_t) \tag{18}$$

Where *θ_c* is the current angle of the UAS. The pitch and roll outputs are held at the midpoint to enable the hover mode of the A2 flight controller. This lets the flight controller make the corrections necessary to keep the UAS at its location. If the height setpoint is reached or the voltage drops below 14.8V, the UAS will continue to the next function.

Once the height setpoint is reached, the UAS changes to the *travel to* function. This function uses the same equations to determine the throttle and yaw outputs, the error distance from the setpoint, and needed direction of travel relative to the UAS. To determine the pitch and roll of the UAS, they are calculated using its GPS location based on its orientation and latitude and longitude error. The latitude error (*La_e*) and longitude error (*Lo_e*) in meters are calculated according to the following equations:

$$La_e = 111195(La_s - La) \tag{19}$$

$$Lo_e = 90745(Lo_s - Lo) \tag{20}$$

La$_s$ is the latitude setpoint, *Lo*$_s$ is the longitude setpoint (the takeoff location), *La* is the current latitude, and *Lo* is the current longitude. The error distance (D_e) from point to point is then calculated by the following equation:

$$D_e = \sqrt{{La_e}^2 + {Lo_e}^2} \tag{21}$$

The direction of travel relative to the UAS (θ_D) is calculated based on the latitude and longitude error, and current angle of the UAS (θ_c) by the following equation:

$$\theta_D = \theta_c - \left(\tan^{-1}\frac{Lo_e}{La_e}\right) \tag{22}$$

Due to the characteristics of the UAS, a derivative term needed to be added to correct overshoot. The future latitude (La_f) and longitude (Lo_f) values were determined by the following equations:

$$La_f = La_0 - D(La_{-1} - La_0) \tag{23}$$

$$Lo_f = Lo_0 - D(Lo_{-1} - Lo_0) \tag{24}$$

Where La_0 and Lo_0 are the current latitude and longitude, and La_{-1} and Lo_{-1} are the previous latitude and longitude, and *D* is the derivative gain. These values are then used in the distance and angle calculations. This error distance (times five) and the direction of travel are converted to rectangular coordinates to determine the amount of pitch and roll, in percent. If the UAS gets within 2.5 meters of the desired location or the voltage drops below 14.6V, the UAS will advance to the next function.

Once the desired location is reached, the *survey* function starts. This function uses the same equations to determine the throttle while the pitch and roll are held at the midpoint to trigger the hover mode of the A2 flight controller. To survey the area, the yaw percentage is held a constant 25%. The UAS maintains this operation while rotating in place. First, the UAS must rotate at least 15 degrees from where it started, and then continue rotating until its less than 5 degrees from where it started to meet the continue criteria. If the UAS completes its rotation or if the voltage drops below 14.6V, it will advance to its next function.

Once the *survey* algorithm is complete, the UAS function changes to *travel back*. The *travel back* algorithm has the same functionality as the *travel to* function, except the setpoint is changed to the latitude and longitude location saved from takeoff. If the error distance becomes less than 2.5 meters or the voltage drops below 14.4V, the UAS will advance to the *land* function.

To land the UAS, the logic is like the *takeoff* function, except the height setpoint is decreased each iteration by a lower rate set by the operator divided by 20. The pitch and roll are held at the midpoint to activate hover mode unless the camera finds a match to the landing fiducial as described below. The UAS continues to lower until it is less than 0.15 meters from the ground, then the outputs go to zero and it returns to the *sit* function.

4.5. Camera Imaging

To ensure the UAS could land onto the UGV with enough accuracy for the guides to align it, camera imaging was used to locate the circles on the landing platform of the UGV. To save computation time and reduce the chance of the camera algorithm finding a false match on the ground, the algorithm was given a size range for which to search. The range of sizes of both circles as seen by the mounted camera as the distance away from the target increases was experimentally determined. This was done by aiming the LIDAR-Lite and camera at the landing platform while changing the distance between them. The distance vs size relationship for the smaller circle was plotted to find a line of best fit. The result is shown in Figure 19.

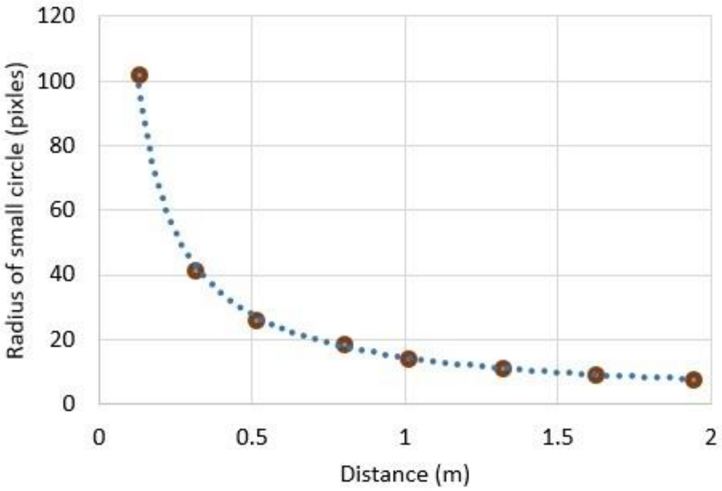

Figure 19: Size versus distance relationship for the small circle

As the UAS gets further away from the platform, the radius of the circle follows a specific curve. Creating a line of best fit, the distance raised to a power had the highest R^2 value of over 0.99, with the second highest being an exponential curve with R^2 being around 0.86. the only type of line that fit the data. In result, the radius of the small circle follows the equation below:

$$S = 14.436D^{-0.941} \tag{25}$$

Where D is the distance between the UAS and landing platform. The R^2 value for this line of best fit was 0.9987, indicating a very good fit to the dataset.

Similarly, the distance versus size relationship for the larger circle was plotted to find a line of best fit. The result is shown in Figure 20.

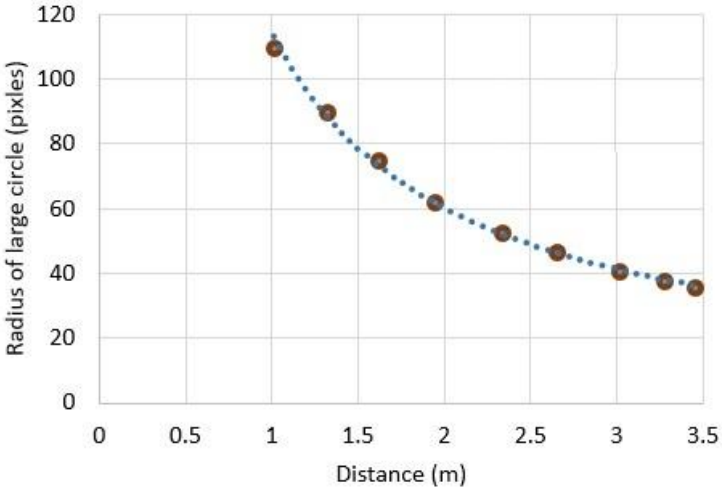

Figure 20: Size versus distance relationship for the large circle

Again, the radius of the circle versus distance follows a specific relationship. Due to the size of the larger circle, it cannot be seen when the UAS is close but does not disappear as easily at further distances. The radius of the large circle follows the equation below:

$$S = 114.44D^{-0.926} \tag{26}$$

Where D is the distance between the UAS and landing platform. The R^2 value for this line of best fit was 0.9984, indicating a very good fit to the dataset.

The UAS needed to decide which of the circles it would search for based on its height. With the size versus distance relationships found, it was decided that the UAS would search for the larger circle when above 2 meters from the ground, and the smaller circle below 2 meters. This gives the UAS a field of view significant enough to find both circles. In practice, the matchable size of the circles needed to cover a significant range due to the latency of the camera processing compared to the velocity of the UAS as it ascends and descends. To compensate, the minimum circle size was set to 50% while the maximum size was set to 150% of the calculated value.

To enhance processing speeds, the webcam was set to capture images 352 pixels wide and 288 pixels tall. With this setting, the processing rate of images was approximately 5 Hz. The camera was set to 30 frames per second to get clear images, but only the most recent image was used for processing; any missed frames were discarded. The first step in processing was to extract a grayscale image using the red plane. A search was performed for ellipses within a certain size criterion. For diagnostic purposes, the processed image was displayed on the operator panel and an ellipse was drawn where it found the highest scored match. Sample resulting images from various distances are shown in Figure 21.

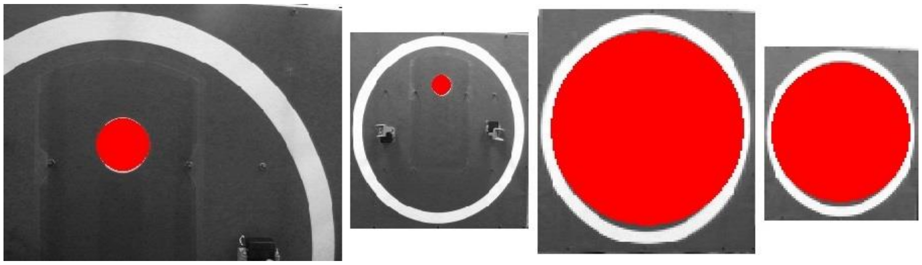

Figure 21: Found circles at 0.49m (left), 1.44m (mid-left), 2.44m (mid-right), and 3.45m (right)

The central position (in pixels), major radius, and minor radius of the best match is used to determine the position of the UAS in relation to the UGV. Since the position is given in pixels, half the width (176) and height (144) of the frame (pixels) had to be subtracted from the position of the match to align the coordinate systems. If the small circle was found, the following equation was used to find the *x* and *y-offset*s (pixels) from the camera:

$$x = S(P_x - 176) \tag{27}$$

$$y = -S\left(P_y - 144\right) \tag{28}$$

Where P_x and P_y are the x and y-position (pixels), and S is the score (%). Since the y-position is counted from the top of the frame and positive pitch tilted the UAS forwards, the y-position had to be negated. Since the large circle is not directly under the camera, the y-position gets adjusted in proportion to the radius of the circle. Nominally, the camera sits 71% of the way from the center of the circle along the radius (*r*), the *y-offset* (pixels) from the larger circle is found by the following equation:

$$y = -S\left(P_y - 144 - 0.71r\right) \tag{29}$$

To create pitch and roll outputs based on the *x*- and *y-position* of a match, the algorithm would apply a proportional output according to the following equations:

$$P = \frac{G * y}{100} \tag{30}$$

$$R = \frac{G * x}{100} \tag{31}$$

Where *P* and *R* are the pitch and roll percentages, respectively, and *G* is the proportional gain. To help the stability and overshoot of the UAS, the hover mode of the A2 flight controller was used whenever possible. For this, if the error was below 40 pixels, the pitch and/or roll were set to 0%. For the UAS to make small corrections for each match, the pitch and roll outputs were reset to zero after 100 milliseconds until a new match was found. A maximum pitch and roll of 25% were also used to prevent overshoot.

CHAPTER 5. DATA COLLECTION

To collect numerical data on the UGV and UAS systems, USB hubs were installed to allow for the use of a USB flash drive while maintaining the use of USB sensors. An algorithm was written to take local variables and record them during a test. A file path is built every quarter-second formatting the file name using the system identifier (UGV or UAS), date, and time; this scheme allows the operator to determine specific time the dataset started. Once the UGV starts the *travel to* function or when the UAS starts the *takeoff* function, it records the current value of the desired variables every 50 milliseconds, indexing them for each iteration. This continues until the control algorithm returns to the *sit* function. It then builds an array from the indexed values and writes the array to a spreadsheet on the flash drive. The algorithm then repeats the first step until the control mode leaves the *sit* function again. Recorded variables could be changed as desired, but data such as control mode, GPS location, height, and orientation were selected.

To comply with UAS regulations set by the NCDOT and FAA, the testing location was determined to be recreational field 12 on the campus of The University of North Carolina at Charlotte, which was surfaced for rugby. The field was located at 35.3045 degrees latitude and -80.7414 degrees longitude.

In addition to numerical data, a Mobius 2 video recorder was used to collect visual data. The camera is capable of 1060p at 60 fps but was constrained to either a horizontal or 30-degree viewing angle since the gimbal was used to stabilize the gimbal.

CHAPTER 6. RESULTS AND DISCUSSION

6.1. Test Settings

The UGV and UAS include parameters such as control multipliers and operation speeds. The adjustable parameters for the UGV include the latitude and longitude setpoint, LIDAR enable and stop distance, and maximum speed of travel. The LIDAR stop distance was set to 0.25 meters. This parameter was determined by manually commanding the UGV to drive towards an obstacle on a tile surface without any avoidance functionality. The stop distance was adjusted until the UGV would reliably stop without any contact with the object. The A2 version of the RPLIDAR does not function reliably outdoors. Due to this, the LIDAR was disabled during outdoor tests. If the newer RPLIDAR A3, which is usable in an outdoor environment, could have been available, this would not have been an issue. The maximum speed of the UGV was set to 100% allowing full speed operation.

The adjustable parameters set for the UAS included the latitude and longitude setpoint, height setpoint, the derivative height gain, rise and lower rates, maximum travel speed, derivative GPS gain and the proportional gain for the camera imaging. The height setpoint was set to 5 meters, with a derivative gain of 20, and the ride and lower rates were set to 0.4 meters per second. The maximum speed was set to 30%, the derivative gain for the GPS was set to 4, and the proportional gain for the camera imaging was set to 15.

6.2. UAS Hover Test

To help understand how well the A2 flight controller can hold its position in hover mode and how well the myRIO can hold a set height and angle, a hover test was performed. For this, the UAS was commanded to take off to and hold its position, 5 meters above the ground, until the two 7Ah batteries were depleted. The myRIO actively corrected for height and orientation while the pitch and roll were held at 0% to enable hover mode on the A2 flight controller. The test was performed with 11 mph winds and gusts up to 20 mph.

The height of the UAS throughout the flight is shown in Figure 22.

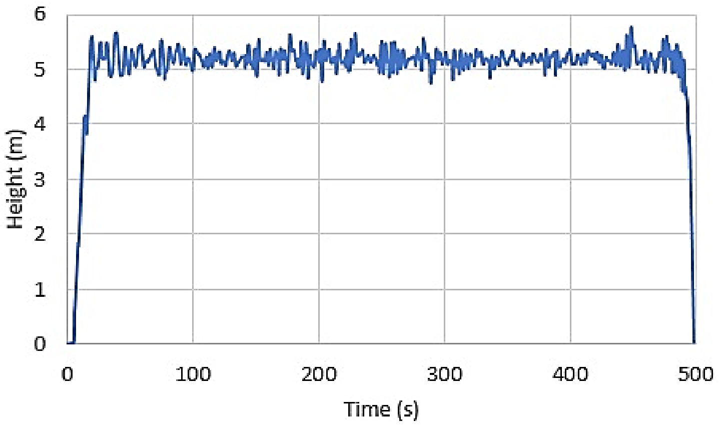

Figure 22: Height of UAS throughout hover test

The UAS took off and held its height for the duration of the flight before landing. Ignoring the first and last 15 seconds for the takeoff and landing, the standard deviation of the height was 0.14 meters with an average of 5.19 meters. While the deviation was not very large, the average was more than one standard deviation above the target. This could be due to the dynamics of the flight controller or an error when calibrating the PWM ranges of the flight controller.

The orientation of the UAS throughout the flight is shown in Figure 23.

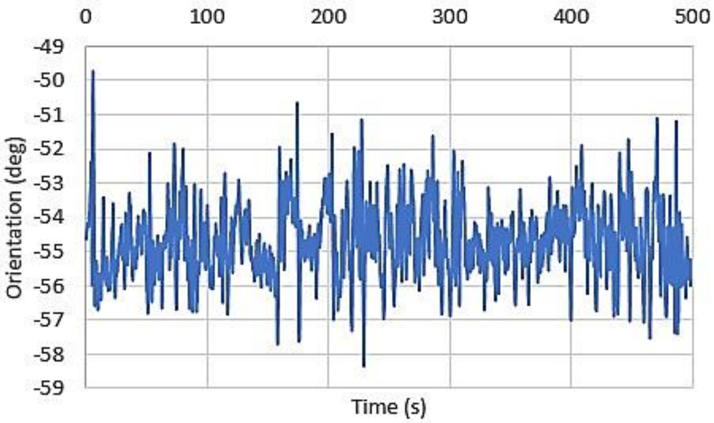

Figure 23: Orientation of UAS throughout hover test

It was noted that the UAS would repeatedly yaw in one direction as it left the landing platform, as seen at the beginning of the plot. However, the UAS did repeatedly correct for it. The standard deviation from this test set is 1.04 degrees. Due to the high cross winds, the turbulence from the propellers can cross each other. With the addition of the response speed of the motors, this can create conditions that are difficult to react to.

While the hover mode on the A2 flight controller held the UAS close to its takeoff position, the myRIO recorded the GPS data from the GlobalSat unit to monitor the drift in the sensor by setting the starting position as the center of the coordinate system. The GPS data from the flight is shown in Figure 24.

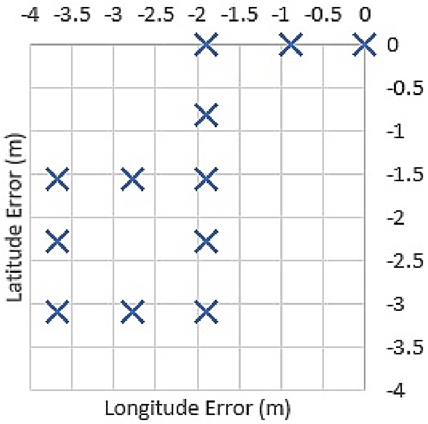

Figure 24: Perceived location of UAS during hover test

Although the GPS and IMU in the flight control system made corrections to keep the UAS in relatively the same position throughout the test, the GlobalSat GPS that the myRIO uses for navigation showed a significant discrepancy. The GlobalSat GPS also made incremental changes of around 0.89 meters in latitude and 0.82 meters in longitude. This may be due to rounding errors internal to the GPS unit but results in a precision about the size of the landing platform. However, its position could be anywhere within a 0.89x0.82-meter square. If it is in one corner during takeoff and the opposing corner during landing, the GPS could calculate the same position although it is 1.21 meters away.

In addition, the distance from the takeoff position was recorded to show drift, as shown in Figure 25.

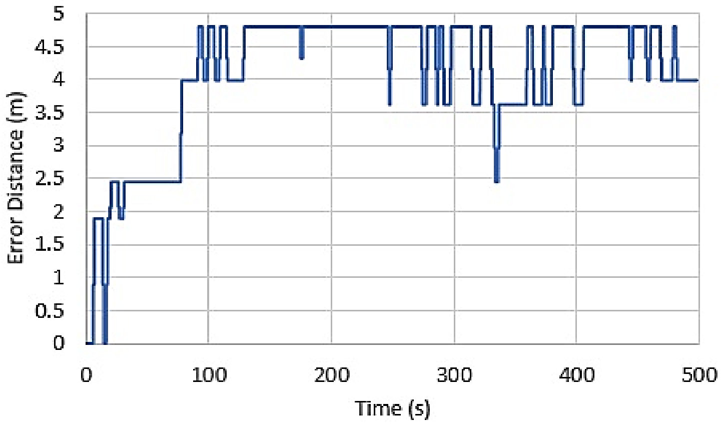

Figure 25: Perceived distance drift of UAS GPS during hover test

As the UAS hovered, the GPS data shows a drift in position. If it is assumed that the UAS stayed stationary throughout the flight, the average drift was 4.07 meters while the maximum was 4.79 meters. Because this error may exist during takeoff and landing, then the GlobalSat unit can only be trusted within a 9.58-meter radius. If it is assumed that the

UAS did change position throughout the flight, it did land on the landing platform, around 0.25 meters away from its takeoff position, as shown in Figure 26.

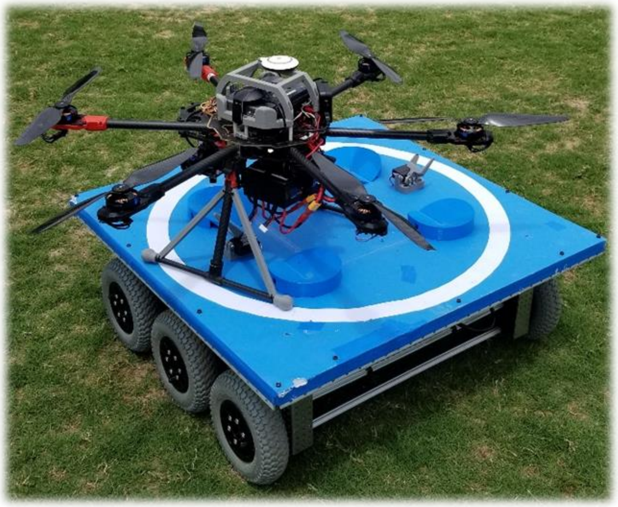

Figure 26: Landing position after hover test

At the time the UAS landed on the landing platform, the GPS calculated it was 3.98 meters away. It is also worth noting that the unit had 10 or 11 out of the possible 12 satellites in view thought out the duration of the flight. This shows the GlobalSat GPS would be unable to land the UAS or get reliable position data throughout flights. Although the GPS position is seen to drift over time, the drift during a short time period would become small. This allows the GPS to show movement in position although it is unable to be used for an absolute position.

6.3. Travel to Location

To test the functionality of the system, it was set to perform a set of tasks. First, the UGV would travel to a location, then the UAS would take off, travel to a different location, survey the area, return and land on the UGV, and then the UGV would return to its start position.

For the first task, the position data of the UGV, with the starting position set as the center of the coordinate system, is shown in Figure 27.

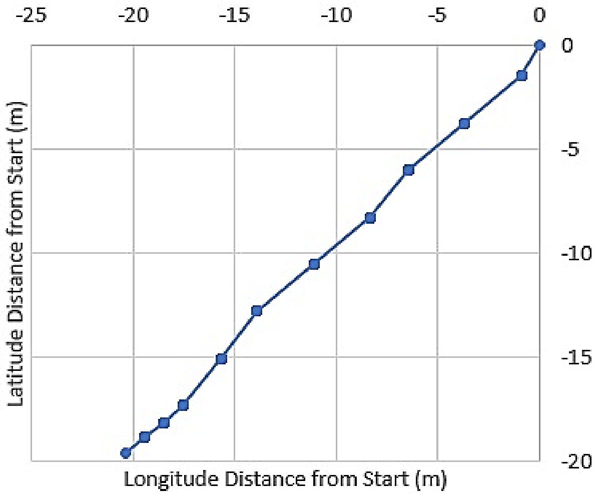

Figure 27: A perceived path of the UGV while traveling to a setpoint

The UGV traveled towards the setpoint and managed approximately 3.4 meters per second. Along the way, its course stayed relatively steady and towards the setpoint. In addition, the distance from the starting position while traveling to the setpoint is shown in Figure 28.

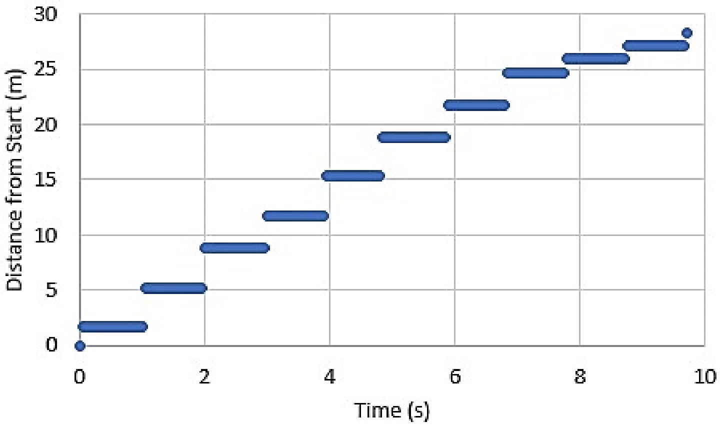

Figure 28: Perceived UGV distance from start location while traveling to setpoint

As the UGV gets close to the destination, the proportional throttle control slowed down until it determined it was within 2 meters of its setpoint and stopped. It is also worth noting that the GPS position changed 50 milliseconds after the UGV received the start command. This is before the UGV could have reacted, indicating the start position could have drifted before the UGV started to move.

The angle of the UGV was recorded and is shown in Figure 29.

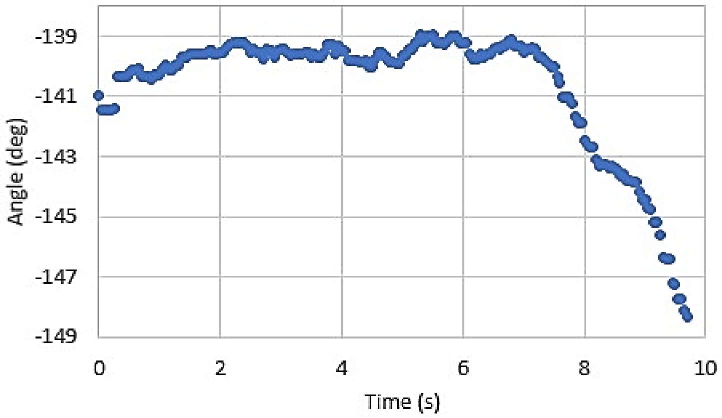

Figure 29: UGV orientation traveling to setpoint

The UGV started at -141 degrees and did not need to adjust far to travel to the destination. As the UGV got closer to the destination, it did correct for an error. This error could have been the precision or drift in the GPS sensor or a calibration error, although the correction was not significant in comparison to the precision of the GPS unit.

Once the UGV arrived at the setpoint and stopped, the servos holding the UAS to the platform released and the UAS started its tasks. The UAS took off like it did in the hover test and preceded to travel to the desired location. The position data during the flight, with the starting position set as the center of the coordinate system, is shown in Figure 30.

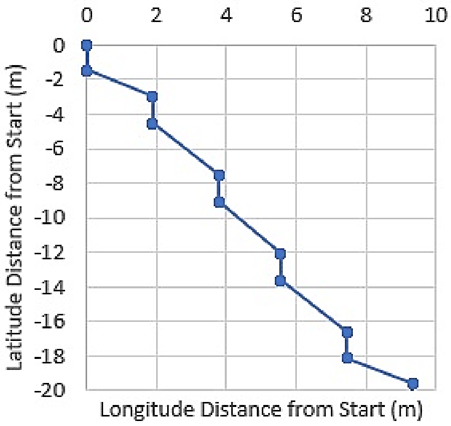

Figure 30: A perceived path of UAS traveling to a setpoint

The UAS traveled towards the setpoint and managed approximately 1.6 meters per second. Along the way, its course stayed relatively the same and towards the setpoint although the positions seem staggered, with each increment being about twice the apparent precision of the GPS from the hover test. In addition, the distance from the starting position is shown in Figure 31.

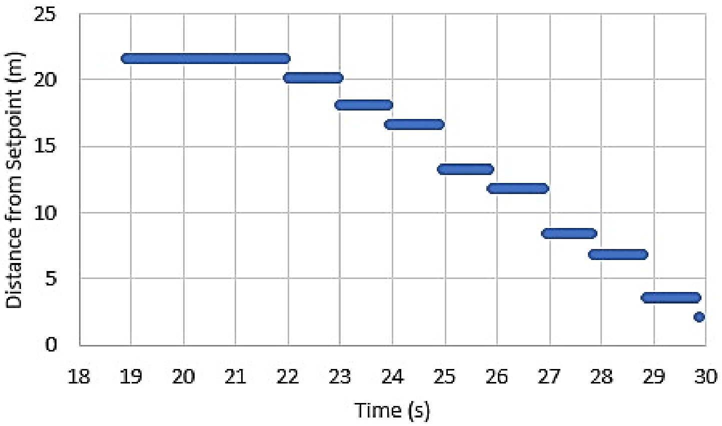

Figure 31: UAS distance from the setpoint while traveling

As the UAS starts to travel, it pitches and rolls towards the setpoint but does not immediately accelerate due to the system dynamics for acceleration. Once the UAS starts traveling, it continues at a relatively constant velocity. The UAS calculated it was 2.12 meters away from the setpoint when it continued to the survey function.

6.4. Survey Area

During the survey function, the 25% yaw rate resulted in the orientation shown in Figure 32.

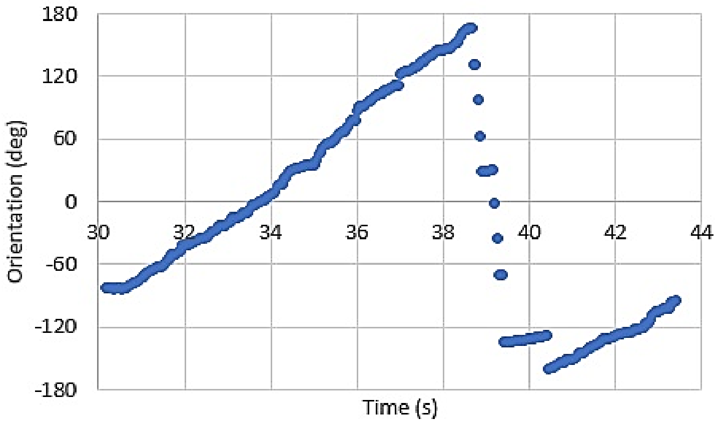

Figure 32: UAS orientation throughout a flight

As the UAS rotates in the clockwise direction, it maintains a rate of approximately 10.7 degrees per second. Once the orientation nears 180 degrees, noise and/or interference create values on either side of the positive to negative 180-degree range. An uncertainty is present due to the positive and negative numbers being averaged. Once within positive or negative 160 degrees, the orientation continues as normal until within the 15-degree criteria.

Position data was also recorded during the survey function and is shown, with the starting position set as the center of the coordinate system, in Figure 33.

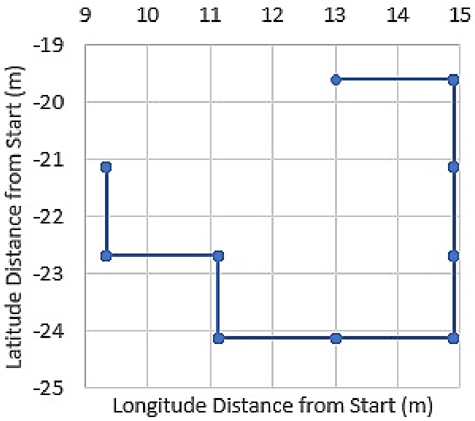

Figure 33: A perceived position of UAS during a survey function

According to the GPS data, the UAS started at approximately 9.3 meters in longitude and -19.6 meters in latitude from the starting position. Although the A2 flight controller used hover mode to keep the UAS in the same position during the survey function, the GPS shows movement. The latitude changed between 24.14 and 19.6 meters while the longitude changed between 9.34 and 14.9 meters from the start position. In this case, the latitude changed by up to 5.56 meters and the longitude changed by up to 4.54 meters. Some drift in position was observed in the last couple seconds of the function, although not significant. Given this information, it is difficult to know where the UAS is due to the errors in the GPS sensor.

6.5. Return to Starting Point

Once the UAS completed the survey function, proceeded to travel back. The position data during the return function, with the starting position set as the center of the coordinate system, is shown in Figure 34.

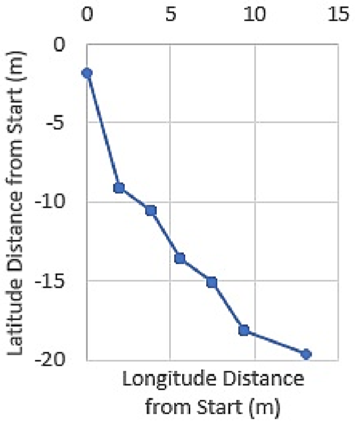

Figure 34: Perceived path of UAS while returning to start point

As the UAS travels back to its original location, its path is mostly towards to setpoint but shows some error in direction. This may be due to factors such as wind or compass errors, although the errors in the GPS unit are more likely. The UAS continues until it is within the 2.5-meter radius of where the UGV should be.

The distance from the setpoint (takeoff location) while returning to the UGV is shown in Figure 35.

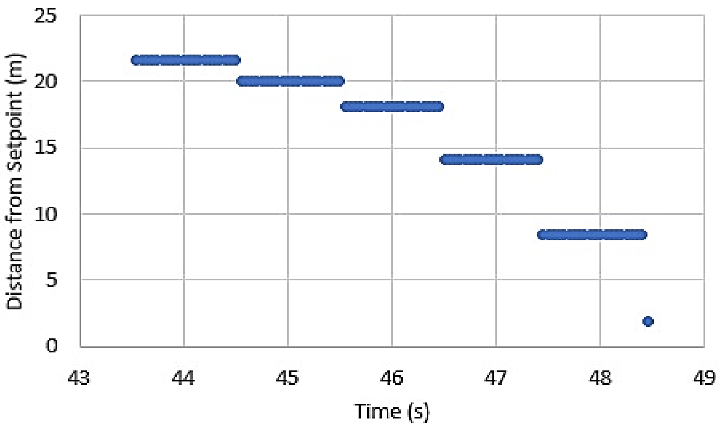

Figure 35: Distance from setpoint while returning

As the UAS banks towards the setpoint, it continues to accelerate. In this case, the UAS appears to keep accelerating until it calculated it was 1.85 meters away from its original location and continued to the landing routine.

As the UAS proceeded to land onto the UGV, it was unable to find the fiducial on the landing platform and therefore did not make any corrections during landing. Although the GPS unit gave information saying the UAS was 1.85 meters away, the next GPS update showed it was 3.48 meters away. During the landing, the error drifted, with the maximum error being 16.8 meters from the UGV without the UAS changing locations. The measured position of the UAS after landing was around 10 meters from its takeoff position, preventing the UAS from finding the UGV. The GPS unit was not able to get the UAS close enough to the UGV during repeated tests, each time having similar results. If the camera had a wider field of view or the UAS flew at a higher altitude, it is possible that the camera would have seen the landing platform but did not happen during testing. It is also worth noting that the UAS maintained its orientation as it did in the hover test thought-out the flight, apart from the survey function.

Once the UAS landed, the UGV was disabled to attach the UAS before it continues to return to its starting position. The GPS data from the return function, with the starting position set as the center of the coordinate system, can be seen in Figure 36.

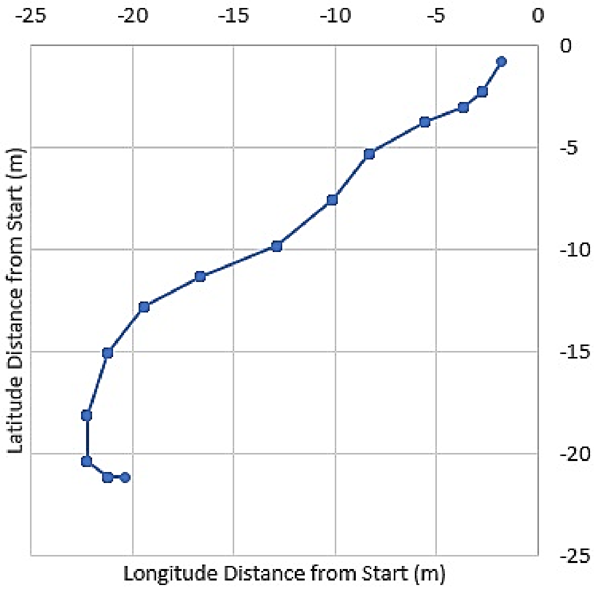

Figure 36: Perceived path of UGV while returning to start position

Since the UGV stopped while facing away from the starting point, it needed to change directions to travel back. The UGV decided to turn right to travel back to the start point, as seen by the partial ellipse shape. Along the way, the GPS data shows some angle oscillations but was not noted during testing. The distance from the starting location is shown in Figure 37.

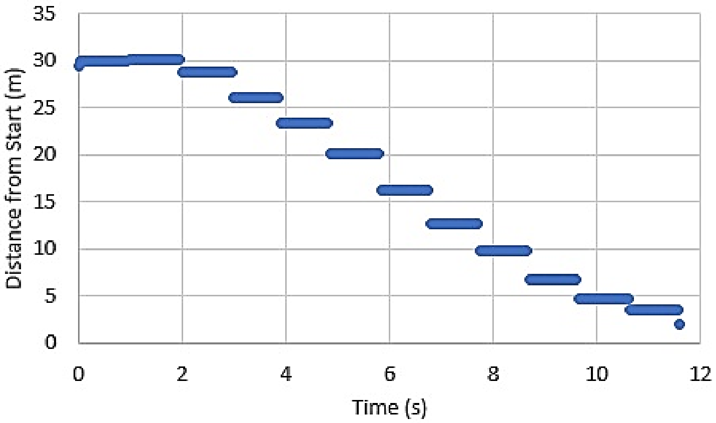

Figure 37: Distance from the starting location while returning as perceived by the UGV

Due to the UGV going forward and changing directions to travel towards the start position, the distance initially increases. Once the UGV is heading towards the start point, the distance decreased at a relatively constant rate. Once the UGV got closer, the speed decreased before stopping 1.92 meters away from start point.

The orientation data recorded while traveling back to the start point is shown in Figure 38.

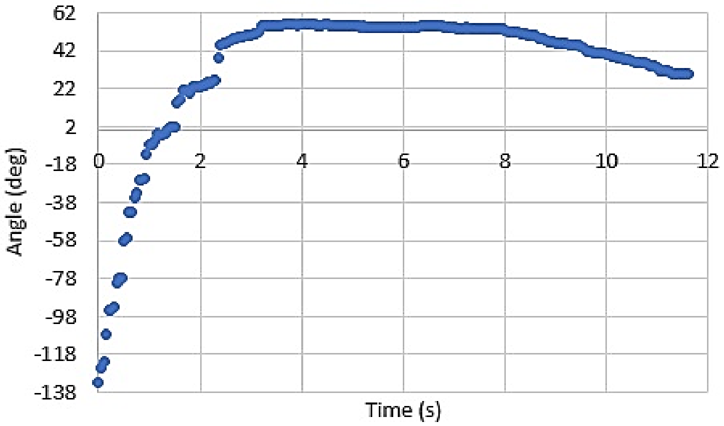

Figure 38: UGV orientation returning to start position

The UGV started at -133 degrees and rotated approximately 188 degrees to travel to the start point. The angle oscillations seen in the GPS data did not appear in the orientation data, meaning it was likely the GPS unit creating them. As the UGV got closer to the destination, it did correct for another error in the same direction as when it went to the setpoint. This may show a calibration error in the compass but is difficult to conclude due to the GPS unit.

Due to the precision and drift seen in the GPS unit, it was determined that the GPS unit alone was inadequate for the UAS to find the landing platform. Without an IMU unit to compare the GPS data to, there was no way to figure out if the GPS data was due to movement in the platform or drift and could not be compared to track GPS position changes. In addition, the UAS would not be able to reliably fly to the desired location and return to the landing platform within a close enough range to land. With this information, tests were created for the UAS to take off and land without traveling to or from locations. It was also noticed that during the hover test, the GPS units incremented its latitude by 0.89 meters in latitude and 0.82 in longitude; however, the GPS increments appear to be double the size during the other tests.

6.6. Takeoff and Landing Tests

During the takeoff and land tests, the UAS was set to rise to a height between 3 and 5 meters, depending on the trial. In adjusting the proportional gain, it was determined that a gain of less than 10 was insufficient to make the necessary corrections while a gain of 18 or higher made corrections too large, sending the UAS off the side of the landing platform far enough that the camera would no longer be able to see the fiducial and make corrections. While a proportional gain of 15 seemed to work the best, it was not able to consistently make the necessary corrections needed. When the UAS had a higher altitude, it could make corrections to stay over the platform without much issue. However, as the UAS came within 1 to 2 meters of the platform it was not able to consistently correct with enough magnitude without overshoot. Adjusting program parameters like the duration and amount of pitch and roll for corrections were made, but the parameters mentioned at the end of the system integration section gave the best results.

Despite the difficulties in landing, the takeoff and landing tests usually resulted in the UAS landing either partially on the platform or close enough for the propellers to contact the platform. The best result is shown in Figure 39.

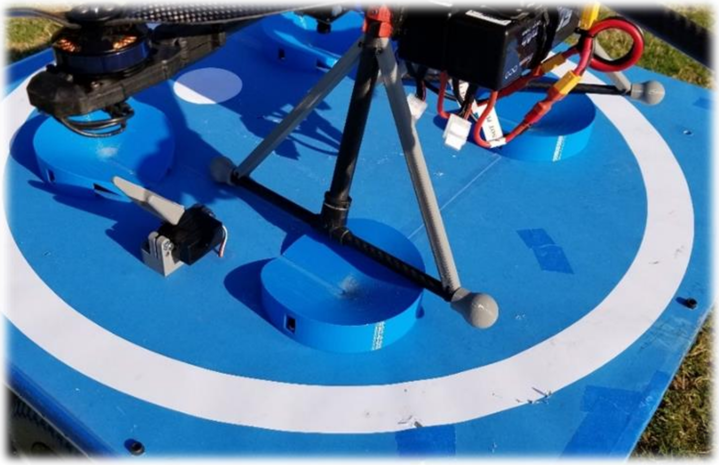

Figure 39: Best result landing UAS onto UGV

It this case, the UAS was approximately 5 centimeters from landing inside the guides, although it was not repeatable. Nevertheless, if the landing platform was twice the length and width, the UAS would have been able to land onto the platform consistently. From there, a system could have developed to push the UAS into the center or hold it in position while traveling. In addition, if IMU data was available, the UAS would be able to calculate how far it needed to correct, track of how far it has moved, then confirm if it was aligned with the platform during landing.

To get the UAS to land onto the platform, a 120-degree camera was installed under the gimbal as shown in Figure 40.

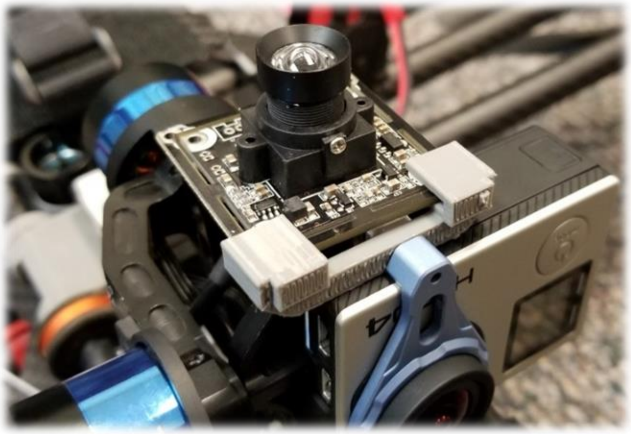

Figure 40: Mounting of 120-degree camera

The camera was mounted underneath the gimbal by a part that was designed and 3D printed. This allowed a GoPro camera to be mounted in the intended spot for video recording. This also allowed the gimbal to pan and tilt as intended, although the tilt had a limited degree of motion. The camera was set to a resolution of 640x480. This reduced the update rate to about 1.5 Hz but gave a similar pixel density along with a 260% field of view compared to the original Logitech camera. In testing this camera in sunlight, the camera was not able to distinguish between the blue and white areas on the platform. Instead, the post-processed camera image appeared as shown in Figure 41.

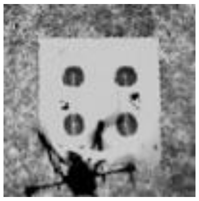

Figure 41: 120-degree camera image of the platform

In this case, it is clear why the program could not distinguish between the blue and white areas of the fiducial. Instead, the platform looks solid white apart from the landing guides. By extracting the green and blue plains instead of the red, the fiducial continued to look white, indicating that there may have been too much light being reflected.

To combat this issue, a polarized filter and different color fiducials were tried. The other fiducials included one with the white areas and perimeter changed to black, and one with the blue changed to black, creating a black and white fiducial. Additionally, the black and blue fiducial was tried while extracting the blue plain to make the blue area appear white. The resulting images are shown in Figure 42.

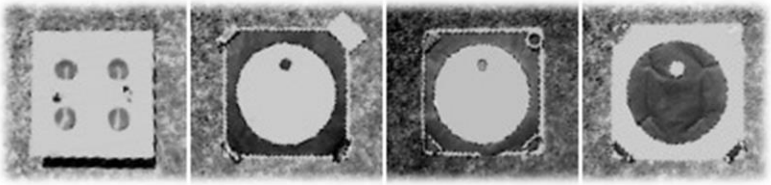

Figure 42: Images (from left to right) with the polarized filter, blue and black fiducial, blue and black fiducial (blue plane), and the black and white fiducial.

While there is likely a lens filter that would allow the camera to view the fiducial correctly, a polarized filter did not correct the problem. The black circles and perimeter did give the camera the contrast needed for the circles to be well defined, although the blue resulted in a nearly white color. Looking at this fiducial while extracting the blue plane was thought to give more contrast and result in a more defined fiducial, however, the black areas were shown as a lighter color while the blue area remained nearly white. While the black and white fiducial was thought to create the highest contrast of any color combination, the black areas were not darker than in the blue and black image while the white was not noticeably brighter. This shows that color combinations may depend on the camera and need to be determined accordingly. In this case, the blue and black fiducial showed the most well-defined fiducial.

To test with the blue and black fiducial, it was held to the ground to the ground with weights while the UAS was placed in the center. The offset to the center of the image was changed to 320 pixels in the x-direction and 240 pixels in the y-direction to correct for the change in resolution.

The UAS then performed the takeoff and landing routines with the 120-degree camera. The results seemed more consistent since the UAS could correct overshoot. While the wider camera angle did have a positive impact, the result but not good enough to repeatedly land the UAS onto the platform.

CHAPTER 7. CONCLUSIONS

Combining an autonomous UGV and UAS could be a versatile solution for surveillance purposes by overcoming the drawbacks of relatively short flight times. The use of a UGV allows the system to travel further and carry larger energy sources allowing the UAS to be recharged or refueled. In addition, the UAS allows for versatile surveillance around, over, and under objects. In this system, the UGV travels to and from setpoints (within a 2-meter tolerance) with its landing platform holding the UAS. Once in position, the UGV releases the UAS which travels to a pre-determined location and surveys the area. Upon return (within a 2.5-meter radius) the UAS attempts to land back on the UGV’s platform. In testing, the prototype’s GPS unit did not have accurate enough position data for the UAS to land correctly on the UGV (an issue likely correctable with the addition of an IMU). When the UAS managed to find the UGV, it made corrections using a camera to align itself with the UGV. As the UAS came closer to the ground, position movements became more critical. Without an IMU to track position changes, it was not able to consistently land onto of the UGV with the 30x30 inch platform. These limitations and potential solutions are presented in section 7.1 below. Alternatively, if the UGV had a landing platform with doubled length and width, the UAS would likely have been able to consistently land onto the platform.

The system showed promising results and could function within the limits of its sensors. A system such as this could also be used for security applications such as temporary military bases or short-term open-air events like the Tour de France. It could also be adapted to inspect solar and wind farms as well as monitor heat using thermal imaging cameras.

7.1. Limitations and Future Work

With the system described above, the precision and drift seen in the GlobalSat GPS unit meant the GPS unit alone was inadequate for the UAS to find the landing platform. Without an IMU unit to compare the GPS data to, there was no way to figure out if the GPS data was due to movement in the platform or drift and could not be compared to track GPS position changes. In addition, the UAS would not be able to reliably fly to the desired location and return to the landing platform within a close enough range to land. The GPS unit used was not able to detect more than 11 satellites during testing; units do exist that can find and calculate position based on 24 satellites. A unit that could find and use up to 24 satellites would have increased the precision of the position data while reducing drift. Additionally, IMU data could have been used to compare to and filter position data for more accuracy. With calculations using 24 satellites and IMU data, it is likely the UAS would have been able to position itself well enough to see the UGV using a camera.

Due to the system dynamics while using the DJI A2 flight controller, the UAS was unable to land onto the UGV with enough accuracy for the system to hold the UAS to the platform. While a wide-angle lens helps in the case of disturbances, an IMU and a more direct control of the position would allow for more accurate changes. If the changes in position were accurate enough, the UAS would be able to land onto the UAS an engage a system to hold the UAS to the platform and/or recharge it.

While using ellipses for the search criteria for the camera imaging, it was possible for the algorithms to find miscellaneous circles on the ground. This would need to be considered based on the environment the system was used in. At one point during testing, the grass field reached a condition where patches of grass did not grow. The areas of ground without grass were occasionally able to be identified as an appropriately sized ellipse. Round objects, such as manhole covers, could potentially cause an issue with the current imaging method. Fiducials with more distinguishable features and/or colors could be used to prevent this issue. Certain color and filter combinations could also be used to eliminate effects of sunlight or enhance the certainty of detecting the UGV.

During testing, wind did not create an issue while using the A2 flight controllers' hover mode. This would lead to the conclusion that wind would not be a large problem if an IMU could be used to make corrections in motor speeds. However, high winds could become an issue. Air traveling across the UAS can send turbulent air across the propellers and create unpredictable conditions that would be difficult for a controller to make corrections to stabilize the UAS.

To extend the operation time of the system, a recharge system would need to be created. Once the UAS lands onto the UGV and gets aligned by gravity or an active system, the energy source for the UAS could be attached to replenish the system as described in other works.

The system as described could be programmed to travel to or between multiple GPS locations. An array of locations could be set and the UAS and/or UGV could repeat through the cycle, going to the next location each time until the energy source is depleted or until it is done. To improve orientation tracking of the UAS and UGV, the uncertainty in angle around 180 degrees should be addressed. The magnetic field for each axis should be averaged before calculating the orientation to prevent positive and negative numbers from being averaged together. The system could also be made to operate from a tablet.

REFERENCES

Brummelen, Glen Van. *Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry*. Princeton University Press, 2013.

Calculate Distance and Bearing between Two Latitude/Longitude Points Using Haversine Formula in JavaScript. https://www.movable-type.co.uk/scripts/latlong.html. Accessed 3 May 2018.

Cheung, Carol, and Benjamin Grocholsky. *UAV-UGV Collaboration with a PackBot UGV and Raven SUAV for Pursuit and Tracking of a Dynamic Target*. Vol. 6962, International Society for Optics and Photonics, 2008, p. 696216. *www.spiedigitallibrary.org*, doi:10.1117/12.777188.

Cocchioni, F., et al. "Autonomous Navigation, Landing and Recharge of a Quadrotor Using Artificial Vision." *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 418–29. *IEEE Xplore*, doi:10.1109/ICUAS.2014.6842282.

Fyfe, W., and R. Johnson. "Unmanned Tactical Air-Ground Systems Family of Unmanned Systems Experiment." *ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005.*, 2005, pp. 103–08. *IEEE Xplore*, doi:10.1109/ROMAN.2005.1513764.

Gruber, J., and A. Anvar. "An Unmanned Surface Vehicle Robot Model; for Autonomous Sonobuoy Deployment, and UAV Landing Platform." *2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*, 2014, pp. 1398–402. *IEEE Xplore*, doi:10.1109/ICARCV.2014.7064520.

HD Webcam C525 - Logitech Support. http://support.logitech.com/en_us/product/hd-webcam-c525/specs. Accessed 3 May 2018.

Kemper, F. Paulo, et al. "UAV Consumable Replenishment: Design Concepts for Automated Service Stations." *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1–4, Jan. 2011, pp. 369–97. *link.springer.com*, doi:10.1007/s10846-010-9502-z.

Li, J., et al. "A Hybrid Path Planning Method in Unmanned Air/Ground Vehicle (UAV/UGV) Cooperative Systems." *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, Dec. 2016, pp. 9585–96. *IEEE Xplore*, doi:10.1109/TVT.2016.2623666.

Moseley, Mark B., et al. *Integrated Long-Range UAV/UGV Collaborative Target Tracking*. Vol. 7332, International Society for Optics and Photonics, 2009, p. 733204. *www-spiedigitallibrary-org.librarylink.uncc.edu*, doi:10.1117/12.820289.

NMEA Data. http://www.gpsinformation.org/dale/nmea.htm#nmea. Accessed 23 May 2018.

Padgett, Steven T., and Aidan F. Browne. *Vector-Based Robot Obstacle Avoidance Using LIDAR and Mecanum Drive*. IEEE, 2017, pp. 1–5. *CrossRef*, doi:10.1109/SECON.2017.7925312.

Petrovic, T., et al. "Can UAV and UGV Be Best Buddies? Towards Heterogeneous Aerial-Ground Cooperative Robot System for Complex Aerial Manipulation Tasks." *2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, vol. 01, 2015, pp. 238–45.

Phan, C., and H. H. T. Liu. “A Cooperative UAV/UGV Platform for Wildfire Detection and Fighting.” *2008 Asia Simulation Conference - 7th International Conference on System Simulation and Scientific Computing*, 2008, pp. 494–98. *IEEE Xplore*, doi:10.1109/ASC-ICSC.2008.4675411.

Rudakevych, Pavlo, and Brian Yamauchi. *A Man Portable Hybrid UAV/UGV System*. Vol. 6561, International Society for Optics and Photonics, 2007, p. 65611B. *www.spiedigitallibrary.org*, doi:10.1117/12.710744.

Steven, D., et al. “Negative Obstacle Detection Using Cooperative Systems.” *ITS World Congress*, 2012, pp. 681–87. *IEEE Xplore*, doi:10.1109/MED.2015.7158825.

William J. Hughes Technical Center. *Global Positioning System (GPS) Standard Positioning Service (SPS) Performance Analysis Report*. Jan. 2017, http://www.nstb.tc.faa.gov/reports/PAN96_0117.pdf#page=20.

Zikou, L., et al. “The Power-over-Tether System for Powering Small UAVs: Tethering-Line Tension Control Synthesis.” *2015 23rd Mediterranean Conference on Control and Automation (MED)*, 2015, pp. 681–87. *IEEE Xplore*, doi:10.1109/MED.2015.7158825.