AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-1053

UAS Operation and Navigation in GPS-Denied Environments Using Multilateration of Aviation Transponders

Christopher W. Lum*, Hannah Rotta†, Ravi Patel‡, Helen Kuni‡,
Tinnabhand Patana-anake‡, Jacob Longhurst‡, Karine Chen‡

Autonomous Flight Systems Laboratory

University of Washington, Seattle, WA, 98195, USA

This paper describes a system for an unmanned aerial system (UAS) to operate and navigate in an environment devoid of a Global Navigation Satellite System (GNSS) such as the Global Positioning System (GPS). The system operates by interrogating an aviation transponder (either mode C or S) that is carried by the UAS and measuring the time elapsed for the response to multiple, ground-based antennas and using triangulation (multilateration) to locate the transponder and by association, the UAS. The ground-based system then routes this position information back to the UAS via the UAS's data telemetry link. The autopilot then utilizes this position information for navigation in much the same way it would utilize a GPS-based position report. This paper focuses on the system architecture to enable a UAS to operate in a GPS-denied environment. Flight test results are presented utilizing a customized version of the popular Pixhawk/ArduPlane avionics platform and demonstrate that the system is capable of guiding a UAS through a series of waypoints in the absence of GPS signals. Furthermore, the customized controller that was designed to consume this alternate source of position information performed well in highly unfavorable environmental conditions. This success illustrates the feasibility of the system as a practical alternative to GPS.

Nomenclature

ADS-B	Automatic Dependent Surveillance - Broadcast
ANPC	Advanced Navigation and Positioning Corporation
AFSL	Autonomous Flight Systems Laboratory
ESC	Electronic Speed Controller
FAA	Federal Aviation Administration
GCS	Ground Control System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IFR	Instrument Flight Rules
ILS	Instrument Landing System
IMU	Inertial Measurement Unit
KDLS	Columbia Gorge Regional/The Dalles Municipal Airport
LAMS	Local Area Multilateration System
NAS	National Airspace System

*Research Assistant Professor, William E. Boeing Department of Aeronautics and Astronautics, University of Washington; Guggenheim Hall Room 211, Box 352400, Seattle, WA 98195-2400, AIAA member.

†Flight Operations Director, William E. Boeing Department of Aeronautics and Astronautics, University of Washington; Guggenheim Hall Room 211, Box 352400, Seattle, WA 98195-2400, AIAA member.

‡Undergraduate Researcher, William E. Boeing Department of Aeronautics and Astronautics, University of Washington; Guggenheim Hall Room 211, Box 352400, Seattle, WA 98195-2400, AIAA member.

PIC	Pilot in Command
PWM	Pulse Width Modulation
RC	Radio Controlled
TCAS	Traffic Collision Avoidance System
TLS	Transponder Landing System
TMP	TRAPIS MAVLink Packager
TRAPIS	TRAnsponder-based Positioning Information System (UW developed)
(s)UAS	(small) Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
UW	University of Washington
UWCUTS	University of Washington Carnation UAS Test Site
WSMP	Washington Simple
WSTR	Washington Steer

I. Introduction

A. Problem Statement

Virtually all unmanned aerial systems (UAS) utilize the Global Navigation Satellite System (GNSS) for navigation and operation. Without access to a GNSS such as the United States' Global Positioning System (GPS), most UAS are unable to operate. In addition to UAS reliance on GPS, the Federal Aviation Administration (FAA) Modernization and Reform Act of 2012[1] outlines steps forward on many aviation fronts to bring aerospace into the modern era. One major initiative from this act is the mandated safe integration of UAS into the National Airspace System (NAS).[2] Another related initiative is the FAA's Next Generation Air Transportation System (NextGen)[3] which showcases the Automatic Dependent Surveillance - Broadcast (ADS-B) system as being a key component for both manned and unmanned traffic.[4] However, a recent investigation by the US Department of Transportation Inspector Generals Office[5] specifically identified these two areas as behind schedule and in need of additional development. This paper focuses on developing technology that addresses these shortcomings and will therefore be vital to the progress of the anticipated $13.6 billion civilian UAS market.[6] ADS-B relies heavily on availability of GPS and cannot function properly without reliable GPS. Further, several elements of the US Department of Defense have repeatedly requested systems and methods which enable UAS operations in GPS-denied situations. This work considers integration and flight testing of an ADS-B equipped small UAS (sUAS) in a GPS-denied environment. It will focus on technical issues associated with integrating a small form factor ADS-B unit with existing UAS avionics. Another major contribution of this paper is the documentation of the operational issues associated with coordinating GPS-denied UAS operations in conjunction with manned aviation within the current FAA regulatory environment.

B. Literature Review

1. Previous Work at the University of Washington

Work at the Autonomous Flight Systems Laboratory (AFSL) at the University of Washington (UW) typically focuses on strategic algorithm development for UAS operation such as search and rescue,[7,8] aerial mapping[9,10] and surveying.[11,12] The AFSL has also investigated situational awareness[13] and human-in-the-loop architectures for UAS operation.[14] Early flight testing work focused on indoor flight testing[15] by collaborating with industry partners, such as Boeing.[16]

2. Related Works

Although previous work in the area of ADS-B implementation on sUAS aircraft is limited, several major studies have been conducted in a similar vein as the research presented in this paper. In 2009, researchers at the University of North Dakota presented software-in-the-loop simulations for an sUAS sense and avoid algorithm which made use of ADS-B information.[17] Another 2009 study involved an ADS-B based collision avoidance system to be used by sUAS in airspace with other unmanned and manned aircraft operating simultaneously.[18] A 2013 study investigated the possibility of incorporating ADS-B transponders on sUAS

and presented a case study to include recommendations for ADS-B regulations regarding sUAS aircraft.[19] More recently, researchers investigated additional sense and avoid algorithms with access to multiple data streams to include traffic collision avoidance system (TCAS) and ADS-B information.[20] While studies such as these have largely focused on future regulations and algorithms for operating sUAS in airspace shared with other sUAS and manned aircraft, the research presented in this paper was focused on demonstrating the use of an ADS-B transponder on a commercially-available sUAS and tracking the aircraft in real time with ADS-B and a secondary Local Area Multilateration System (LAMS) unit for GPS-degraded and GPS-denied operations.

II. System Architecture

This section describes the components, connections, interaction, and function of the various pieces that make up the overall system to allow UAS to operate in GPS-denied environments.

A. Overall System Architecture

This system utilizes a combination of hardware and software components to enable GPS-denied operation. Figure 1 shows a high-level overview of the architecture. Subsequent sections will detail each sub-system in greater detail. Roughly speaking, data packets containing information about the UAS such as GPS coordinates and aircraft type are received by either an ADS-B in receiver (for example a Sagetech Clarity) or the LAMS. These packets are consumed by a custom software application (denoted as TRAPIS 1.0) which processes, filters, and feeds into the TRAPIS MAVLink Packager (TMP) subsystem. The TMP extracts GPS coordinates and altitude of the plane, packages the data, and sends it to the unmanned aerial vehicle (UAV) via a MavProxy link.

B. TRAsponder-based Positoining Information System (TRAPIS)

TRAPIS is the major software application in this system. It serves as the main user interface with the system. The various features, capabilities, function, and architecture of TRAPIS has been documented in previous publications.[21–24] A high-level block diagram of the system is shown in Figure 1.

C. Local Area Multilateration System (LAMS)

The Local Area Multilateration System (LAMS), shown in Figure 2, is a compact surveillance radar developed by the Advanced Navigation and Positioning Corporation (ANPC). LAMS is a split-off portion of the ANPC Transponder Landing System (TLS),[25] which is used for instrument flight rules (IFR) approaches at airports in difficult terrain where traditional instrument landing system (ILS) approaches are impossible. The LAMS and how it is used in this current work was previously documented in other publications.[21,22]

D. ADS-B Payload

The ADS-B transponder used for this project was a Sagetech Corporation model XPS-TR, which is shown in Figure 3(a). Customization of this payload to integrate into an sUAS was discussed in previous publications.[21,22]

The transponder in the previously described configuration received its position information from the aircraft's GPS by connecting into the UAS Pixhawk and GPS unit. For this paper it was desirable to have the transponder payload act independently of the UAS, so instead, a secondary Pixhawk and GPS unit were integrated into the payload. This allows the payload to be easily integrated onto multiple UAS, or even operated without integrating it onto a vehicle at all.

The customized and integrated payload of ADS-B transponder, Radio Control (RC) receiver, and Arduino comprise the primary hardware described in this paper. These components are part of the TRAPIS system and are hereafter referred to as the TRAPIS payload.

Two different views of the payload are shown in Figure 4. Figure 4(a) gives an overall view of the payload and all its connections between components. Figure 4(b) shows one possible configuration of these components, with the transponder located inside the box that houses the Arduino. This configuration was used in much of the early testing of the system, but presented a number of issues. During testing, the

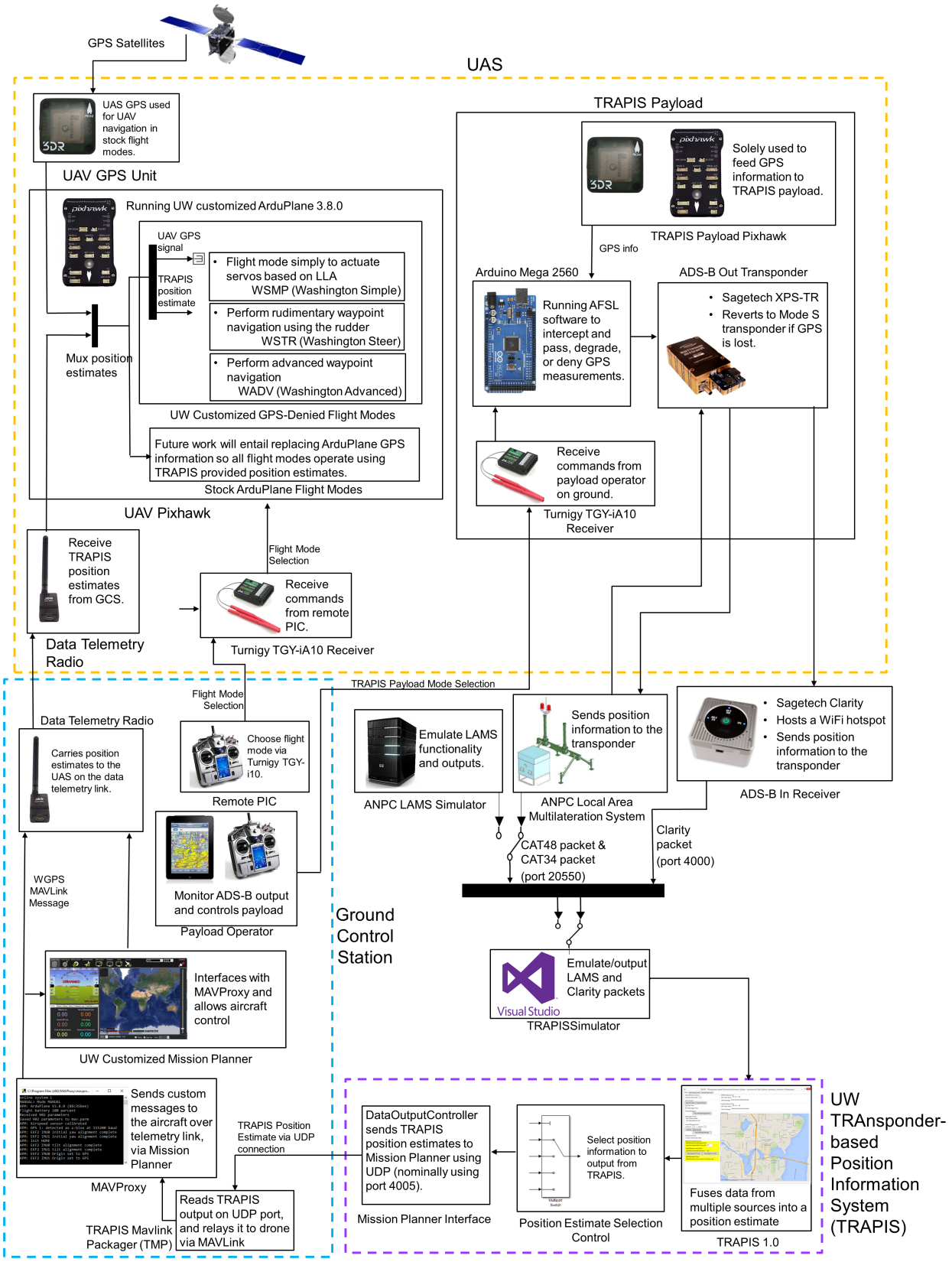


Figure 1. TRAPIS block diagram.

(a) LAMS installed at KDLS.

(b) Interior of a LAMS unit.

Figure 2. LAMS physical setup.

(a) Sagetech Corp. XPS-TR ADS-B out transponder.

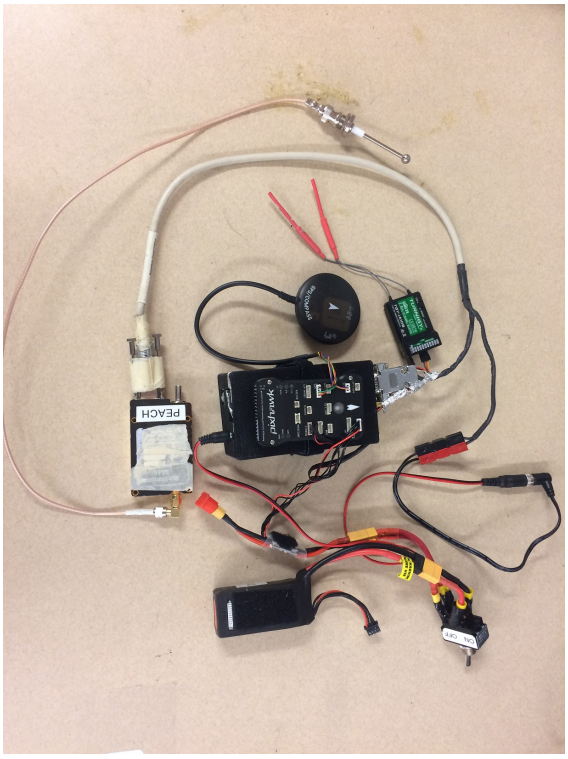
(b) Sagetech Corp. Clarity ADS-B in receiver.

Figure 3. Sagetech ADS-B systems used.

transponder frequently overheated, causing it to shut down and cease transmission. This demanded that it be removed from the box and placed in a less heavily insulated location. This also provided the flexibility required to combat the electromagnetic interference between the transponder and other components of the aircraft. The final configuration that was used in the later stages of flight testing is shown in Figure 5. This setup features the transponder located on the exterior of the aircraft, on the underside of the right wing, with the remainder of the components still located within the aircraft payload bay. This allowed the transponder antenna to reach to a location where it would cause minimal interference, and provided airflow over the transponder unit for cooling without significantly disrupting the aerodynamics of the UAS.

E. Aircraft

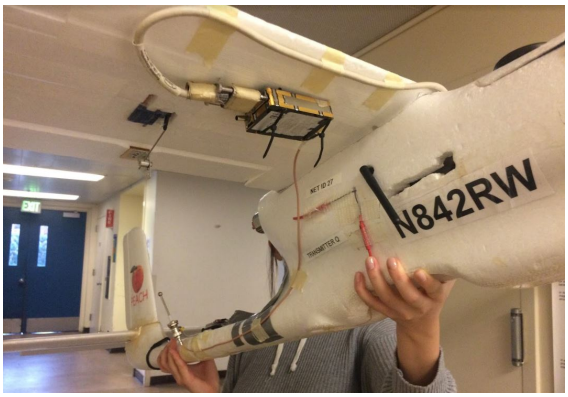
The payload for this experiment was flown on a Finwing Sabre, which is a fixed wing UAS. The aircraft weighs 3.12 kg, including all internal components and the experimental payload. It has a wingspan of 1.9 m and measures 1.32 m from tip to tail, with the center of gravity located 2 inches behind the leading edge of the wing. The aircraft is controlled by a Pixhawk autopilot. This is connected to several other components, including a 2.4 GHz receiver for pilot communication, a 915 MHz Holybro telemetry radio for communication with the ground station, and a Holybro GPS/Compass Module to provide heading and GPS navigation capabilities. This aircraft was selected for its stability, reliability, and payload capacity. Modifications to the airframe were made to accommodate placement of the battery forward of the main payload bay, and to accommodate the size of the payload. These modifications consisted of removing excess



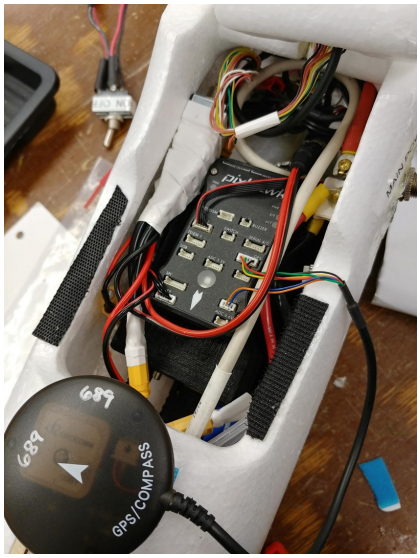
(a) Payload with all standalone components, designed to operate completely independently of the UAS, showing the transponder outside the box.

(b) Interior of the payload showing the Arduino and transponder inside the box. Note that this configuration proved unfeasible and was not used in the final flight test.

Figure 4. TRAPIS payload with integrated transponder, Arduino, Pixhawk and GPS units.



(a) The transponder's final location on the underside of the wing.



(b) The Arduino, payload Pixhawk, and payload GPS located in the aircraft payload bay.

Figure 5. The final configuration of the TRAPIS payload.

material from the airframe to create the necessary space. A full list of the internal components is shown in Table 1.

	Components	**Comment**
Guidance, navigation and control	Pixhawk with ArduPlane firmware	See reference 26
Flight Control Surfaces	Aileron (2 servos on 1 channel)	Turnigy^{TM}TGY-50090M Analog Servo MG 1.6 kg/0.08 sec/9g
	Rudder (1 servo on 1 channel)	
	Elevator (1 servo on 1 channel)	
	Flaps (2 servos on 1 channel)	
Propulsion	Motor	Turnigy^{TM}D3542/5 1250 KV Brushless Outrunner Motor
	Propeller	APC LP09060E Thin Electric Propeller (9x6 in.)
	Battery	Turnigy^{TM}5000mAh 3S 25C LiPo Pack
	ESC	Turnigy^{TM}TRUST 45A SBEC Brushless Speed Controller
	Thrust	1350 g
Communication	Standard 2.4 GHz RC transmitter/receiver combination	Turnigy^{TM}TGY-i10
	915 MHz telemetry transmitter/receiver combination	Holybro^{TM}500 mW power
GPS Navigation	GPS/Compass Module	Holybro^{TM}Ublox NEO-M8N
TRAPIS Payload	Arduino Mega 2560	See Figures 3 and 4
	Sagetech Corp. XPS-TR ADS-B out transponder	
	Sagetech Corp. Clarity ADS-B in receiver	
N-number & ICAO	N842RW (AB88D3) w/ squawk code 1253	

Table 1. Major sUAS components used for experiment.

F. TRAPIS MAVLink Packager (TMP)

The TRAPIS MAVLink Packager (TMP) takes location data generated by the TRAPIS application and relays it to the UAV using MAVLink protocol. The TRAPIS application encodes a TRAPIS packet into a JSON encoded string. TRAPIS then sends that JSON string to a specified port using UDP. TMP is a python script that:

1. Reads the encoded JSON string from TRAPIS at a local network port using UDP.
2. Decodes the packet and extracts the latitude, longitude, and altitude information.
3. Transmits the information as a MAVLink message that can be detected and read by the UAV's firmware.

The TRAPIS packet data structure follows the same structure as the standard MAVLink COMMAND_LONG message as shown in Table 2. TRAPIS uses command ID 999 for the "command" field. This ID must be unique to any other MAVLink command. Command ID 999 is not used by any other standard commands, so it is available for TRAPIS to use. This allows the UAV's firmware to address a TRAPIS message uniquely. TMP takes the decoded latitude, longitude, and altitude from the TRAPIS application and assigns them to parameters 1, 2, and 3 respectively. This message is then packaged by calling message_factory.command_long_encode() function defined in the DroneKit Python library, which follows the structure of the COMMAND_LONG message in the XML file. The firmware on-board the aircraft is then able to look for this command ID and record these values by reading the parameters of this MAVLink message.

Field Name	Type	Description
target_system	uint8_t	System which should execute the command
target_component	uint16_t	Component which should execute the command, 0 for all components
command	uint16_t	Command ID (of command to send)
confirmation	uint8_t	0: First transmission of this command. 1-255: Confirmation transmissions (e.g. for kill command)
param1	float	Parameter 1 (for the specific command)
param2	float	Parameter 2 (for the specific command)
param3	float	Parameter 3 (for the specific command)
param4	float	Parameter 4 (for the specific command)
param5	float	Parameter 5 (for the specific command)
param6	float	Parameter 6 (for the specific command)
param7	float	Parameter 7 (for the specific command)

Table 2. MAVLink COMMAND_LONG structure that the TRAPIS packet follows.

The Dronekit Python library is an open source project developed by 3D Robotics Inc. It provides functions to communicate and command UAVs that are able to comprehend MAVLINK protocol. TMP periodically receives JSON packets from the TRAPIS application and sends the custom message to the UAV. Mavproxy, a command-line ground control system (GCS) that allows connection to the UAV via a telemetry link, is used to connect the script to the UAV. More importantly, it provides multiple bi-directional UDP ports, so that both the TMP and other GCS, such as Mission Planner can connect to the UAV at the same time. Hence, Mavproxy acts like a middleman by passing information from the Python script or Mission Planner to the UAV. For the TRAPIS project, Mavproxy opens two bi-directional UDP ports: one for Mission Planner and one for the TMP script. Overall, TMP packages and sends MAVLINK messages. The sent message is routed to Mavproxy and Mavproxy relays the message up to the UAV.

G. Firmware Modification

The UAV uses custom firmware to navigate using TRAPIS data. The firmware uses ArduPilot version 3.8.0 as the foundation. ArduPilot is an open source autopilot firmware. It comes with several modes that feature waypoint navigation and orbiting around a point. These built-in modes require GPS to fly properly. Using

ArduPilot 3.8.0 as the base for the custom firmware allowed TRAPIS to leverage some of ArduPilot's built-in features while still creating a GPS-denied mode.

The custom firmware has several major components. First, the UAV must understand the MAVLink messages being sent by the TMP. Second, the UAV must do something with the TRAPIS position estimates. The custom firmware includes a custom mode: WSTR. This is a simple flight mode that allows the UAV to use the TRAPIS position estimates instead of GPS data to navigate a flight path by only using its rudder. The flight mode is also discussed in detail below.

1. Consumption of Custom TMP Message

The ArduPilot firmware had to be modified to understand and store incoming TRAPIS MAVLink messages sent by the TMP. The firmware has a list of MAVLink commands that it is willing to receive. It distinguishes between MAVLink messages based on the command ID. For each command ID, the firmware has a set of instructions to execute. TRAPIS uses the MAVLink command **COMMAND_LONG** to send TRAPIS data to the UAV (through the TMP). In Table 2, the COMMAND_LONG MAVLink message has a *command* field. This command field has an integer that represents an ID that corresponds to a set of instructions. Therefore, the COMMAND_LONG message contains a command. The **MAV_CMD_NAV_WAYPOINT** command, represented by command ID 16, is an example of a command that can be contained in the COMMAND_LONG message. The MAV_CMD_NAV_WAYPOINT command is shown in Table 3. Note that the parameters shown in Table 3 correspond to the parameters that would be sent in the COMMAND_LONG message.

Value	Field Name	Description
16	MAV_CMD_NAV_WAYPOINT	Navigate to waypoint.
	Mission Param #1	Hold time in decimal seconds. (ignored by fixed wing, time to stay at waypoint for rotary wing)
	Mission Param #2	Acceptance radius in meters (if the sphere with this radius is hit, the waypoint counts as reached)
	Mission Param #3	0 to pass through the WP, if ¿ 0 radius in meters to pass by WP. Positive value for clockwise orbit, negative for counter-clockwise orbit. Allows trajectory control.
	Mission Param #4	Desired yaw angle at waypoint (rotary wing). NaN for unchanged.
	Mission Param #5	Latitude
	Mission Param #6	Longitude
	Mission Param #7	Altitude

Table 3. MAV_CMD_NAV_WAYPOINT MAVLink message structure

TRAPIS requires a new set of instructions to follow when the plane receives a TRAPIS message. The specific message structure used for TRAPIS is shown in Table 4. The ArduPilot firmware was customized so that it saves the most recent latitude, longitude, and altitude from TRAPIS. The TRAPIS messages are received at approximately 1 Hz, so the firmware is essentially saving the current position of the UAV as seen by TRAPIS. Now that the UAV knows and understands its position without using GPS, it must be able to use that position data appropriately.

The custom firmware also includes a custom flight mode: Washington Steer (WSTR). WSTR was designed to be a simple flight mode that allows the UAV to navigate a flight path using TRAPIS position estimates. WSTR treats the most recent TRAPIS position received as the current position of the UAV. It then compares that position estimate to the current waypoint. WSTR then uses a proportional-derivative controller to adjust its heading towards the desired waypoint heading by only using the UAV's rudder. WSTR holds a constant altitude and zero bank angle so that only the rudder is used for steering. WSTR only uses the rudder because this mode is only designed to prove that a UAV can navigate a flight path using TRAPIS estimates. This custom flight mode is not designed to efficiently navigate those waypoints.

WSTR consists of three separate controllers: the wing leveler, which commands only the ailerons, the steering, which commands only the rudder, and the altitude hold, which commands only the elevator.

Value	Field Name	Description
999	MAV_CMD_TRAPIS	Update current plane position (using TRAPIS data).
	Mission Param #1	TRAPIS-estimated Latitude
	Mission Param #2	TRAPIS-estimated Longitude
	Mission Param #3	TRAPIS-estimated Altitude
	Mission Param #4	Empty
	Mission Param #5	Empty
	Mission Param #6	Empty
	Mission Param #7	Empty

Table 4. MAV_CMD_TRAPIS MAVLink message structure

Throttle is controlled manually by the pilot for simplicity, and to allow the operator to make adjustments for varying wind conditions, as it was known that the final flight test of this system would likely take place in strong or extremely variable winds. As the actual control mechanism operating on the aircraft was not intended to be a major focus of this research, a controller that would take the minimum amount of time to design and tune was desirable. A block diagram showing the overall control scheme is shown in Figure 6.

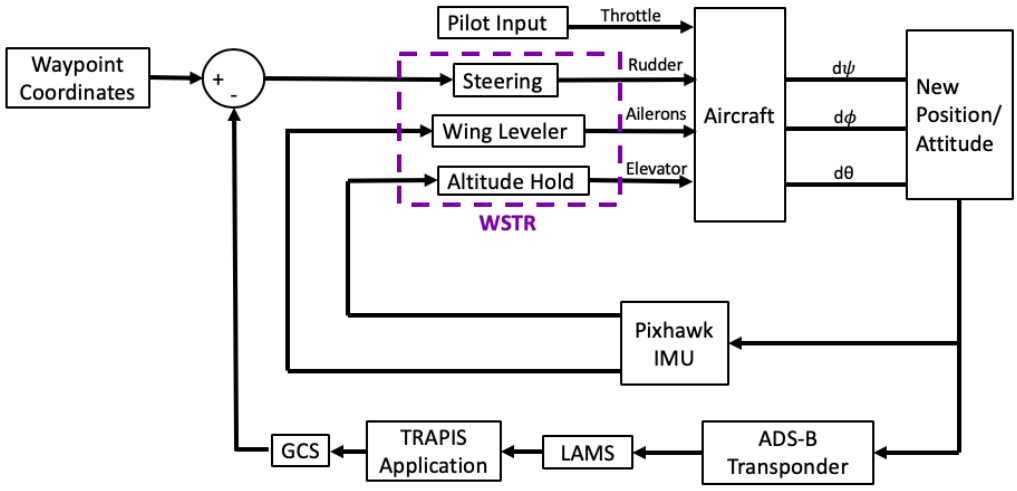


Figure 6. Block diagram showing the flow of signals through the WSTR controller.

Each component of WSTR also imposes a limit on the maximum control surface deflection angle it will command. This served as a safety mechanism to protect the aircraft from abrupt maneuvers in the event of an erroneous control signal. The ailerons, rudder, and elevator were all limited to deflect by no more than $\pm 30°$. The PWM value this corresponds to varies from aircraft to aircraft, but by imposing a limit on the physical deflection angle of the control surface rather than the PWM value, it was ensured that the controller output would always remain within a safe bounded region, regardless of any differences in how the servos were calibrated from aircraft to aircraft.

2. Wing Leveler

Figure 7 shows the signal flow through the wing leveler portion of the WSTR controller. It is a Proportional-Derivative (PD) controller which maps the current bank error and outputs the appropriate aileron deflection angle. As noted previously, this angle is limited to $\pm 30°$.

Figure 7. Block diagram showing the flow of signals through the wing leveler component of the WSTR controller.

3. Altitude Hold

Figure 8 shows the signal flow through the altitude hold portion of the WSTR controller. The altitude error is fed into a Proportional controller which computes the pitch error, which is then fed into a PD controller that outputs an elevator deflection angle. Again, this angle is limited to $\pm 30°$.

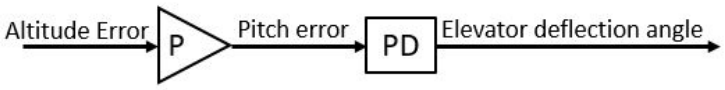


Figure 8. Block diagram showing the flow of signals through the altitude hold component of the WSTR controller.

4. Steering Control

The steering controller, which is a pure proportional controller, maps heading errors between the current location of the aircraft and the desired waypoint location to a rudder deflection. In order to prevent large amounts of overshoot or oscillation in the response of the controller, the gain was intentionally kept very low. This resulted in very gradual maneuvers compared to the behavior of more complex steering controllers, such as ArduPilot's built in Auto mode. Note as well that the steering control does not involve the ailerons in any way. For simplicity, this controller was designed to steer purely using the rudder, as opposed to Auto mode which uses the ailerons for coordinated turns. Figure 9 shows how the heading error angle is computed from a geometrical perspective.

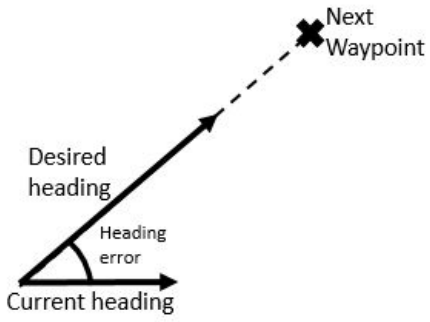


Figure 9. Heading error geometry.

One notable behavior of the steering controller is the fact that when it begins navigating through a new flight path, it is set to begin at waypoint number 2. This was done for convenience, to facilitate the large amount of simulation that was done prior to actual flight testing of the controller. In the ArduPilot simulator, waypoint number 1 must always be designated as a takeoff waypoint, which cannot be taken in by the controller. Therefore the controller was designed to ignore waypoint number 1, to minimize the modifications necessary to transition the controller from simulation to flight testing.

III. Experimental Methods

This section details the significant test campaign that was performed to validate the system.

A. Initial Ground Testing

The ground testing took place at the UW campus and at the UWCUTS location (which is discussed further in the next section) and primarily consisted of testing the TRAPIS payload to ensure that it is capable of supplying a reliable signal that is robust enough to be used for navigation. This included verifying that the aircraft position could be received by the Clarity ADS-B In receiver and displayed on both an iPad running an ADS-B compatible application, WingX Pro, and the TRAPIS application, and that the ADS-B position could be successfully fed back to the aircraft.

A very basic, custom "flight" mode to be used for ground tests only, Washington Simple (WSMP) was used to verify that the custom controllers could command control surface deflections on the UAV based on the payload position information fed back to the UAV via the TMP. WSMP commanded the ailerons to deflect 10 degrees in a certain direction if the UAV was carried to one side of an artificial line defined by GPS coordinates, and deflect the other direction if the UAV was carried to the other side of that line. This validated the ability of the system to command the UAV based on custom position information.

After this success, a more complex flight mode, WSTR, was created to incorporate the wing leveler, altitude and steering controllers. WSTR uses rudder to navigate while the ailerons and elevator keep the wings level and maintain altitude respectively.

It was verified that without using the UAV's on-board GPS, the complete TRAPIS system (using GPS from the TRAPIS payload) could command a UAV in WSTR mode to steer to a series of waypoints located on the ground. To do this, a researcher walked the system to each waypoint and demonstrated that the rudder commanded it to correctly "steer" to the next waypoint.

All the tests described above relied on the transponder being fed GPS position through its standalone GPS receiver. However, the purpose of this project is to be able to navigate without GPS reception at any point in the system. This requires being within the operational range of the LAMS, so to simulate using the LAMS without actually operating within its operational radius, a scheme was devised to be able to "hijack" another aircraft's LAMS position information.

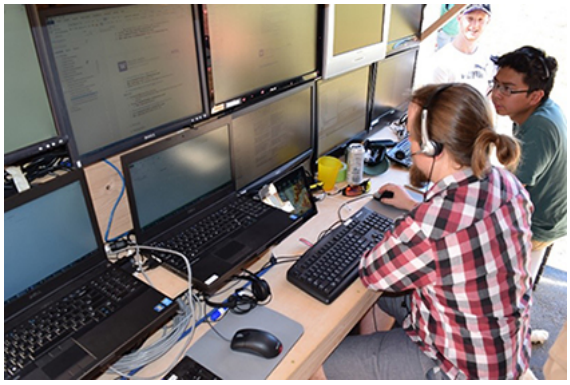
Live aircraft position information was sent from the LAMS, located in Dallesport, Washington, to a computer in the AFSL via a UDP port. The TRAPIS application in the lab could then process actual aircraft data from 150 miles to the south in real time. Position information was retrieved from a manned aircraft in the vicinity of the LAMS. The original goal was to be able to feed a UAS the position from one of these other manned aircraft, and have the UAS navigate based on the LAMS information during flight. If the desired waypoint is located at the UW Carnation UAS Test Site (UWCUTS) 150 miles north of the LAMS-tracked aircraft, regardless of the current position of the UAS, the flight controllers would command the UAS to fly north, because it would think it needs to fly from Dallesport to the Seattle area. Unfortunately, the process to send the live LAMS position information wirelessly to UWCUTS proved impractical. Instead, the test was performed with the aircraft on the ground on the UW campus, where the live LAMS data was being received. This test was completed by verifying that the rudder deflected in the correct direction based on the current heading of the UAS on the ground. The rudder always deflected such that the aircraft would try to fly north.

B. Initial Flight Testing

The initial flight testing took place at UWCUTS, which is approximately 500 acres of open, grass fields located near Carnation, Washington. The payload was integrated into the Finwing Sabre UAV in the configuration discussed previously. One of the aircraft with the payload integrated is shown in Figure 10(a). The Ground Control Station (GCS) was located in an enclosed trailer called the Mobile Flight Operations Center (MFOC), which has been modified by support UAS flight operations. The interior of the MFOC is shown in Figure 10(b). Previous works describe the infrastructure of the MFOC.[21,22]

The goal of the flight testing at UWCUTS was to verify all of the individual components of TRAPIS separately and as a combined system, to ensure that it was fully functional and reliable for the final demonstration in Dallesport, WA. The main components that required verification were the transponder payload, wing leveler/altitude hold flight controller, and the rudder steering controller which also included the TMP.

(a) TRAPIS payload integrated onto the Finwing Sabre UAS, showing secondary GPS unit, and transponder antenna mounted to the empennage.



(b) The interior of the MFOC.

Figure 10. The aircraft and GCS setup used for flight testing.

Table 5 outlines the complete testing campaign that was used to test and demonstrate these systems both on the ground and in the air.

The controllers were tested by first feeding them fully functional, normal GPS position information. Each test was then repeated using the GPS position fed to the GCS via the ADS-B transponder inside the TRAPIS payload. These tests consisted of commanding the UAS to navigate through a series of waypoints in WSTR mode. Several different waypoint configurations were tested, as shown in Figure 11. The Single waypoint (Figure 11(a)) demonstrated the controller could steer in the correct direction. The Simple waypoints (Figure 11(b)) demonstrated that it could handle transitioning from one waypoint to the next without any sharp turns. The Full, or Demonstration waypoints (Figure 11(d)) were designed to be the same size and shape that were actually flown during the final flight demonstration in Dallesport. The Full waypoints were rigorously tested and validated to ensure WSTR functioned well in a variety of conditions, especially with varying wind velocities. The Circle Path waypoints (Figure 11(c)) were originally intended for the final demonstration, and were used for much of the early validation of WSTR at UWCUTS. This path was eventually replaced with grid path shown in Figure 11(d) once it was demonstrated that the controller was robust enough to support a more complex flight path. A grid pattern was selected due to the high demand for similar flight paths in many UAS applications, such as aerial mapping, search and rescue, and agriculture.

ArduPilot's geofence function was also thoroughly tested at UWCUTS. The geofence is a set of predetermined geographical boundaries, which include an altitude limit, beyond which the aircraft cannot go. If the aircraft crosses the geofence for any reason, the autopilot will take over and fly it back to a point near the middle of the geofence, and cause it to loiter in a circle about that point. Multiple tests were conducted where the aircraft was deliberately steered across the geofence in various modes (including WSTR) to verify this functionality. The maximum distance beyond the edge of the geofence the aircraft would reach before turning around was noted, and used to position the waypoints in relation to the geofence at the final flight test.

C. Final Flight Testing

A complete flight test was conducted at the Columbia Gorge Regional Airport (KDLS) on June 12, 2018. Three data runs were conducted over the course of two flights, all using the Demonstration waypoints (Figure 13). For safety reasons,[27] given the proximity to manned air traffic,[28] a geofence (shown as a red polygon surrounding the flight path in Figure 13) was used to keep the aircraft within the testing area, and under 116 meters above ground level. This geofence was in place during all flights.

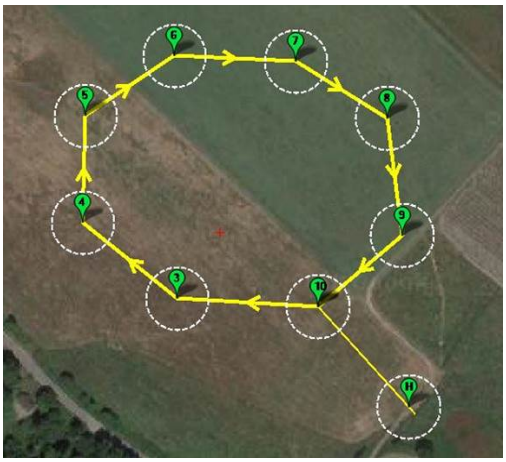
First, the waypoints were flown in ArduPilot's built-in Auto mode to collect baseline data regarding the conditions and the performance of the aircraft when using GPS. This run is referred to as the Auto run. The Auto run also differed from future runs in that Auto mode uses a highly refined and effective inner loop controller that yields very accurate trajectory tracking, unlike WSTR, which only uses the rudder to steer.

Date	Test	Description
02/08/18	TRAPIS payload bench test	- Verify the TRAPIS payload functions in the lab
02/10/18	TRAPIS payload ground test	- Verify the payload functions at UWCUTS
02/23/18	WSMP ground test	- Verify TMP can accept packets from TRAPIS - Verify TMP can send customized packets to WSMP
03/03/18	Dummy payload flight test	- Verify the aircraft can safely carry the payload
03/09/18	WSTR ground test	- Verify steering controller functions with waypoints - Position is based on UAV GPS position, not ADS-B or LAMS output
03/10/18	Payload flight test	- Verify payload functions during flight - Demonstrate that the payload does not interfere with aircraft systems
03/10/18	Steering to single waypoint	- Demonstrate WSTR can successfully steer toward a single waypoint - Use aircraft position information, not ADS-B or LAMS output
03/10/18	Steering to simple flight path	- Demonstrate WSTR can successfully steer through a series of simple waypoints - Use aircraft position information, not ADS-B or LAMS output
03/10/18	Steering to full "Dallesport" waypoints	- Demonstrate WSTR can successfully steer through the full flight path - Use aircraft position information, not ADS-B or LAMS output
03/27/18	WSTR ground test 2.0	- Repeat of previous WSTR ground test, but using ADS-B position output instead of the aircraft's GPS position
04/13/18	Geofence ground test	- Verify the geofence works in custom flight modes even if the GCS loses connection to the ADS-B navigation
05/12/18	Geofence flight test	- Verify the geofence works in all flight modes
05/12/18	Steering to single waypoint with payload	- Demonstrate WSTR can successfully steer toward a single waypoint - Use ADS-B position output
05/12/18	Steering to simple flight path with payload	- Demonstrate WSTR can successfully steer through a series of simple waypoints - Use ADS-B position output
05/12/18	Steering to full "Dallesport" waypoints with payload	- Demonstrate WSTR can successfully steer through the full flight path - Use ADS-B position output
06/11/18	Hijack LAMS data ground test	- Verify aircraft appropriately deflects rudder on the ground "borrowing" live LAMS data being sent from KDLS
05/26/18, 06/08/18	"Dress Rehearsal"	- Run through all the tests for final verification

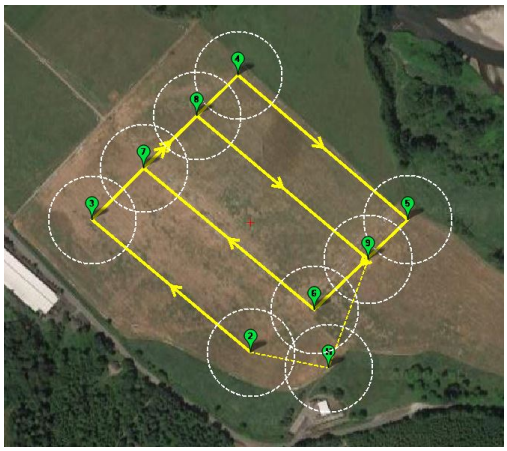
Table 5. Test campaign.

(a) Single waypoint flight path.

(b) Simple waypoint flight path.



(c) Circle waypoint flight path.



(d) Final demonstration "Dallesport" flight path.

Figure 11. Waypoints used in initial flight testing at UWCUTS.

The second run, called the LAMS run, was the GPS-denied run using WSTR mode. This run demonstrated the functionality of the full TRAPIS system, using position data from the LAMS unit routed through the ground station and back to the aircraft. This test demonstrated the use of the entire system in a GPS-denied environment.

The third and final run, known as the ADS-B run, was a second control run designed to fully repeat the tests that were performed at UWCUTS. This test involved feeding data from the ADS-B In receiver into the TRAPIS ground station. Because the ADS-B In receiver uses GPS, this run was not GPS-denied, but it serves to demonstrate the ability of TRAPIS to take in a different source of position information as well as isolate the effectiveness of the WSTR inner loop controller.

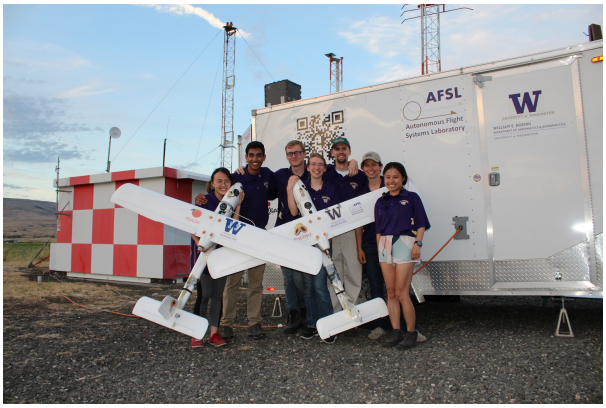


Figure 12. Researchers from AFSL at the final flight test at KDLS, with the MFOC installed in front of the LAMS unit.

Prior to launch, the connection between the aircraft and the LAMS and Clarity receiver was established and it was verified that the aircraft's location was being tracked accurately on both the TRAPIS

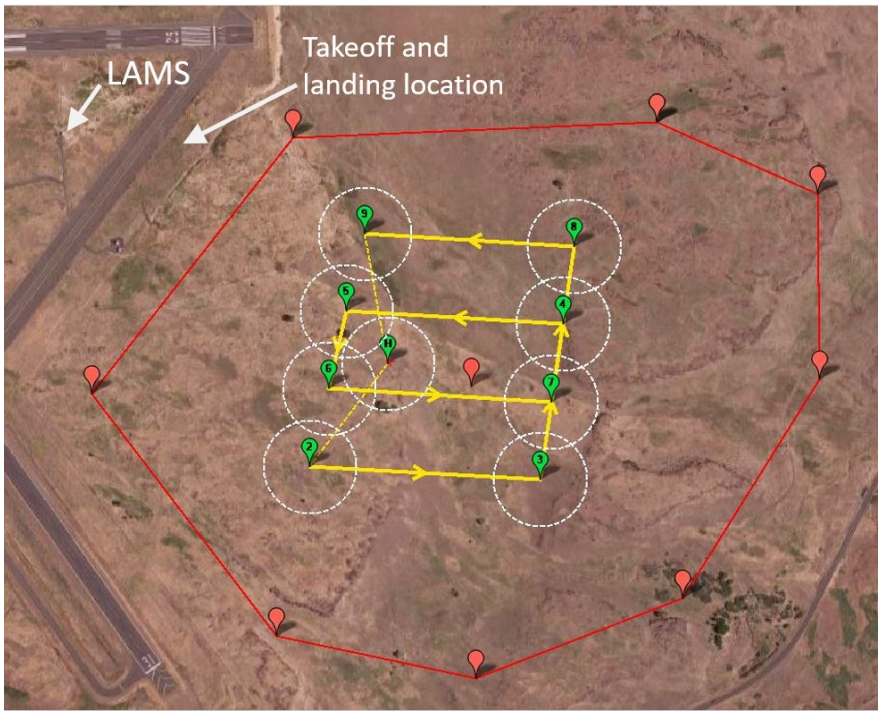


Figure 13. The test site at KDLS, showing the location and orientation of the Demonstration waypoints.

application and the iPad. After launch, the pilot manually climbed to an altitude of approximately 100 meters. Once the aircraft was within the boundaries of the geofence, the geofence was activated. Then the aircraft was switched into Auto mode, then WSTR with the source of position data being controlled by the GCS, located at the LAMS unit. Upon successful completion of the flight path, the geofence was deactivated to allow landing in the proper location and the aircraft was landed manually.

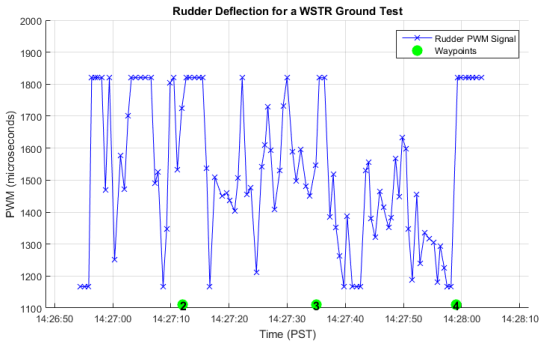
Throughout testing, a second Remote Pilot in Command (PIC) maintained communications with local manned aircraft via an air band radio, announcing UAS activity in the area prior to takeoffs and landings, and instructing the UAS operator to land immediately when manned aircraft were taking off or landing. This operation was coordinated well in advance with the KDLS management to minimize any possible risk to manned aircraft activities.[29]

IV. Results

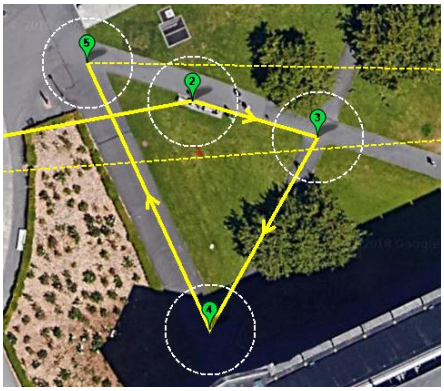
A. Ground Testing

Figure 14(a) shows the Pulse Width Modulation (PWM) signal being sent to the rudder during the WSTR ground test that was conducted on March 9, 2018. During this test, a research carried the aircraft through the waypoints shown in Figure 14(b), changing the orientation of the aircraft so that the rudder was returned to center after every deflection, manually simulating the behavior of an aircraft steering through waypoints. These waypoints consisted of a triangle of right-handed turns (note that the controller begins at waypoint number 2, as discussed previously). For the majority of the test, the aircraft was commanding PWM values greater than 1500, indicating that the aircraft was attempting to make these right-handed turns. This test demonstrated that the WSTR controller was functional using GPS position, and suitable for flight testing.

Extensive ground testing of the TRAPIS payload was also conducted. Much of this testing was aimed at minimizing electromagnetic interference between the transponder and various components of the UAV. This interference, which was also identified as a problem in previous work on this system, is due to the large difference in power between the signals sent by the transponder (5 W) and the signals commanding the control surfaces (0.1 W).[21] These issues were directly correlated to the proximity of the transponder and its antenna to various components, such as the Pixhawk, ESC, and servos. Most notably, the Pixhawk



(a) Rudder deflection.



(b) Ground test waypoints.

Figure 14. Results from a WSTR ground test on March 9, 2018.

autopilot computer consistently failed to record data flash logs while the payload was on. Data flash logs are recorded on-board the Pixhawk and include data such as position information, aircraft system information and other in-flight data. In certain payload configurations, the control surfaces would sometimes move unpredictably, commands would lag, or the motor would fail to spin when commanded. Aluminum foil was initially wrapped around the transponder in an attempt to minimize this interference, but this caused the transponder to overheat and cease function properly, and was therefore removed. Various positions of the transponder and its antenna on-board the UAV were tested, and it was found that the interference could be minimized by fixing the transponder to the underside of the wing, as far away from the body of the aircraft as was possible without jeopardizing stability, with the antenna on the tail of the aircraft. Unfortunately the problem of data flash logs failing to record was not able to be resolved before the final flight test at KDLS. However, telemetry collected by the GCS was able to provide an equivalent set of data.

B. UWCUTS Flight Testing

As described previously, initial flight testing was performed at UWCUTS which unfortunately is not equipped with a LAMS. In order to exercise as much of the system as possible without a LAMS, TRAPIS consumed the ADS-B output of the aircraft, processed this data, and then routed this back to the aircraft. This allowed the aircraft to navigate using ground-based position information (the processed ADS-B data) and demonstrate that navigation via the proposed architecture is feasible. The trajectory of the aircraft during one of these flights, as recorded via telemetry logs with respect to the desired waypoints, is shown in Figure 15, along with the trajectory recorded by TRAPIS. Notice that the aircraft is able to navigate to these waypoints successfully using this ground-based position information, thereby showing that the architecture is viable for GPS-denied navigation. Also note that the TRAPIS position reports are slightly different from the aircraft telemetry due to the fact that this uses a separate GPS receiver.

C. KDLS Flight Testing

At the final demonstration flight test at KDLS, the aircraft was able to navigate successfully through the waypoints using GPS position, ADS-B receiver data, and LAMS data. This last test demonstrated the ability of the system to allow an aircraft to navigate without GPS. The flight path of the aircraft as it navigated the waypoints using LAMS position estimates is shown in Figure 16. Note that the aircraft's actual position (given by GPS) closely matches the LAMS position estimates throughout the flight, and that every waypoint in the flight path was successfully reached. During this test, the GCS operator noted wind speeds of up to 20 knots from between approximately 300° and 310° (true heading), which caused the aircraft to deviate from flying in a straight line between waypoints.

Figures 19(b), 18, and 19(a) show the crosstrack error associated with the three primary data collection runs performed at KDLS. The first used Ardupilot's Auto mode, which uses GPS for navigation. Then followed two runs using the custom WSTR mode, the first being a completely GPS-denied run using position estimates from the LAMS, and the second using position estimates from the ADS-B In receiver (mimicking

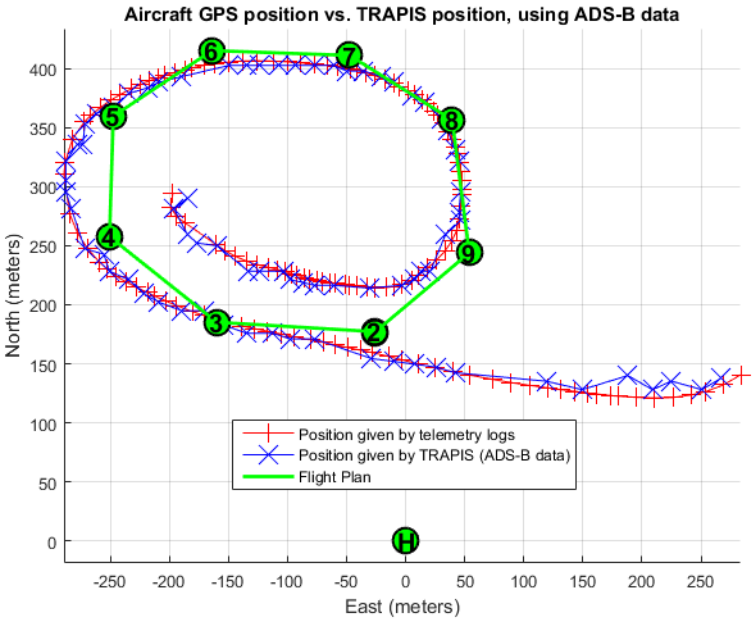


Figure 15. Trajectory and ADS-B data used for navigation during a test at UWCUTS.

the preliminary tests performed at UWCUTS). These three runs all used the waypoints shown in Figure 13. The UAS was considered to have reached a waypoint and began targeting the next one when it got within 75 meters of the waypoint. The times at which this occurred are also shown. Note that by default, WSTR skips waypoint 1, which is why the waypoint numbering begins at 2.

Except for significant deviations from the planned trajectory while the aircraft was targeting waypoints 3 and 7, the amount of crosstrack error during either of the WSTR runs is not significantly greater than the amount displayed by Auto mode (60-80 meters at the most). These deviations can be traced back to two major sources. The first is the wind. During all these tests, there was a nearly constant wind of up to 20 kt. WSTR relies entirely on the rudder for steering, while Auto makes use of the ailerons for coordinated turns. Without the use of the ailerons, the capability of an aircraft to maintain correct heading in severe winds is significantly degraded. The second major source of error stemmed from the transmission of the position data itself. During the run using Clarity position estimates (shown in Figure 19(a)), data points were received sporadically, especially during the parts of the trajectory when the aircraft was furthest from the ground station. This indicates a possible limitation in the wireless range of the system.

Furthermore, during both the Clarity and LAMS runs, there was a delay in the transmission of data, so that the position being used to command the rudder deflection lagged several seconds behind the actual position of the aircraft. This lag was likely due to inherent time delay associated with off-board processing of navigation data. The impact of this behavior was most noticeable when compounded by the lack of any transmission at all for large portions of the Clarity run, most notably between waypoints 2 and 3, and waypoints 6 and 7. During these segments, the aircraft would begin by making a gentle turn, which it would continue to hold until receiving another position information packet. Until then, it would continue to think it was at the location where it received the first packet, and so would hold the same rudder deflection it would need to reach the next waypoint from that location, causing it to continue in a far too gentle turn, steering it in the wrong direction until the next packet was received. It can be seen from Figure 17 that while no data was being received, the aircraft maintained a relatively constant turning radius, as it was holding roughly the same rudder deflection it began commanding when it received its last data point. This can be verified by examination of Figure 21(a), where periods of constant rudder deflection can be observed during the portions of the flight path where no ADS-B data was being received. Wind drift likely also contributed to this phenomenon, carrying the aircraft still further away from where it thought it was, rendering the

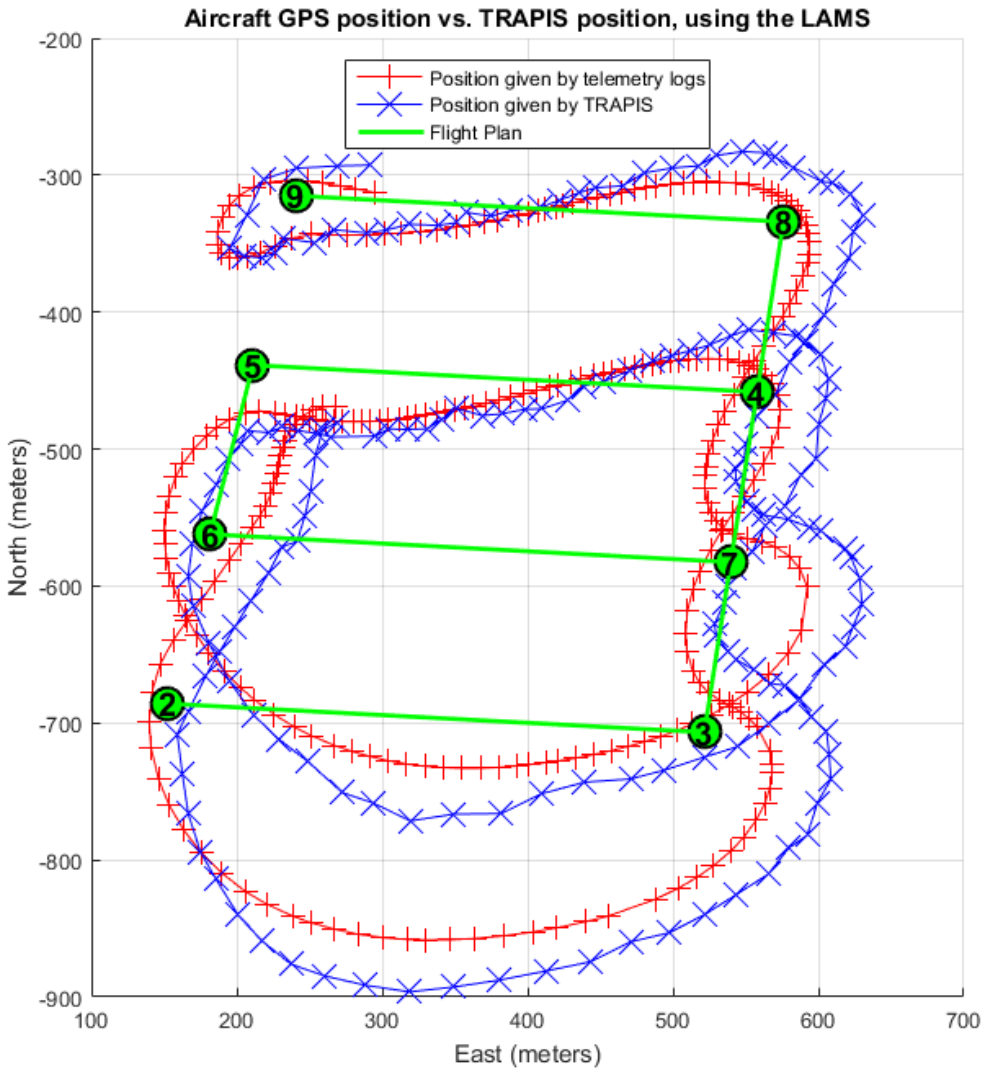


Figure 16. The LAMS navigation data used during the final demonstration, along with the actual trajectory flown by the aircraft using this data.

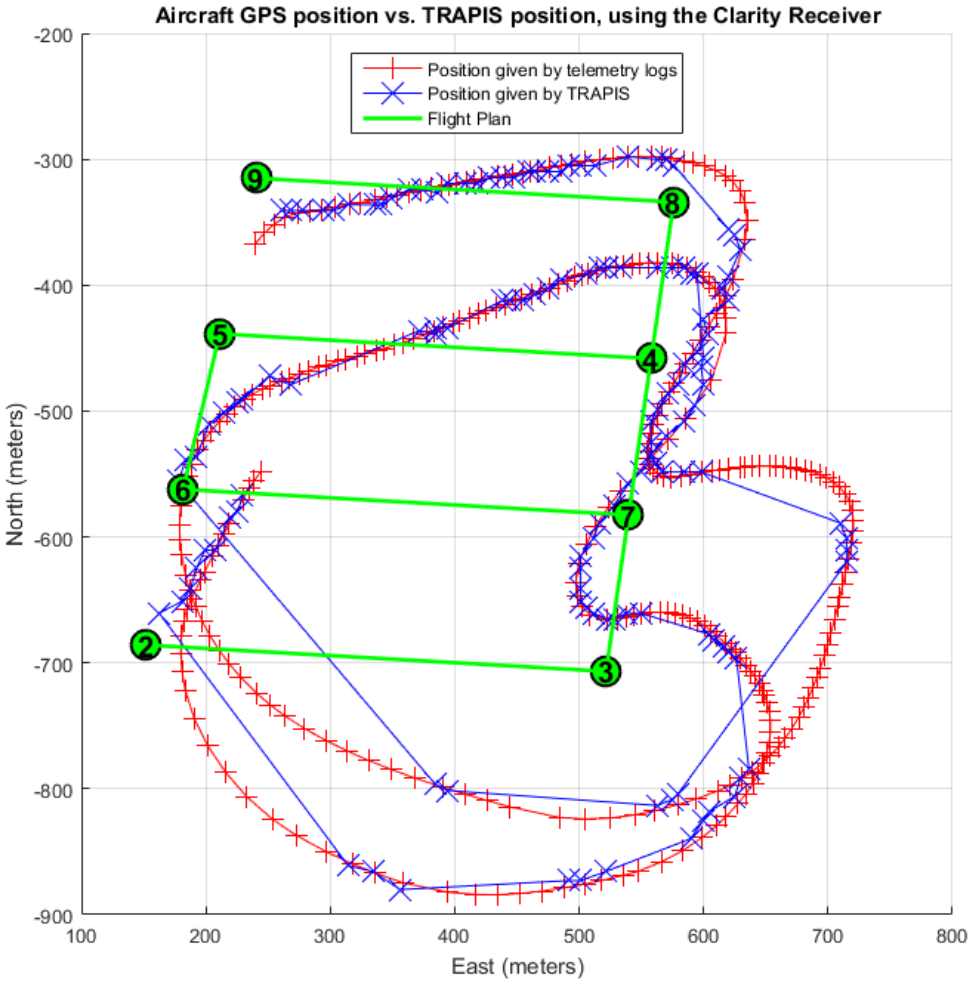


Figure 17. The ADS-B navigation data from the Clarity Receiver used during the final demonstration, along with the actual trajectory flown by the aircraft using this data.

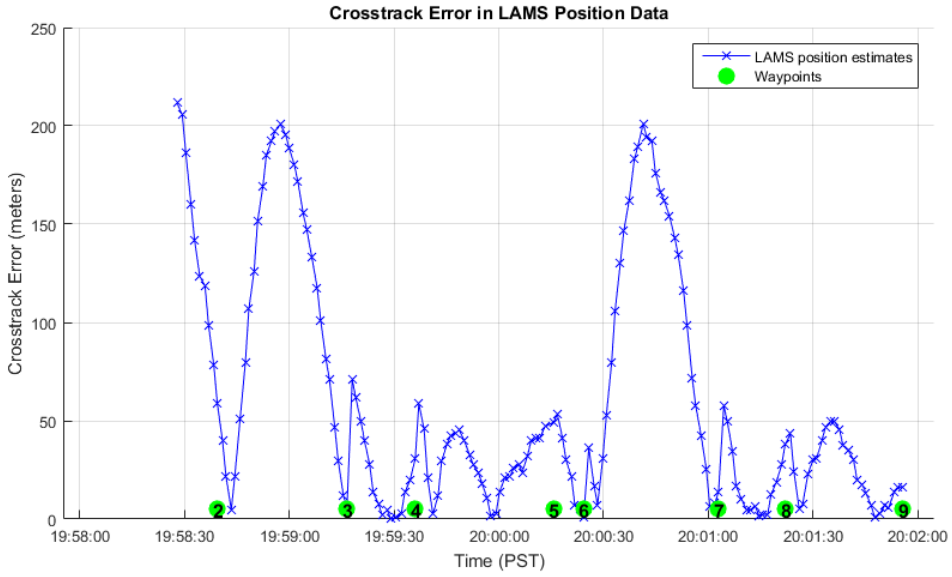
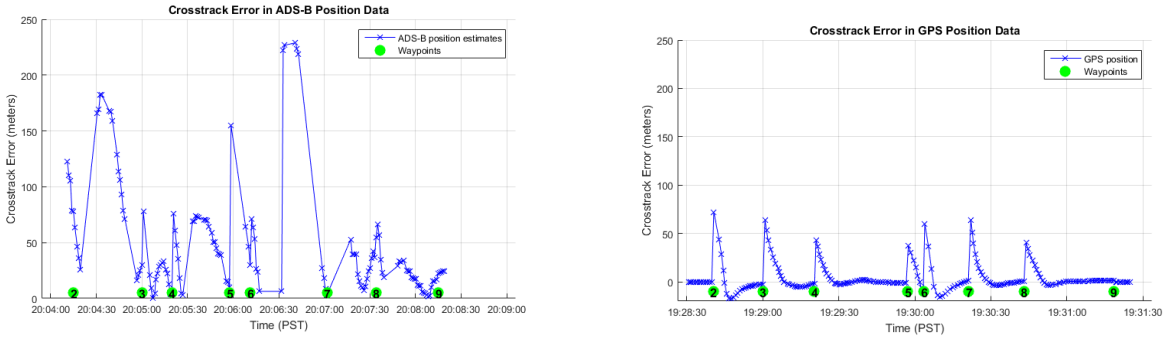


Figure 18. Crosstrack error in position for WSTR navigating through the final demonstration waypoints at KDLS using position information from the LAMS.



(a) Crosstrack error for the ADS-B run.

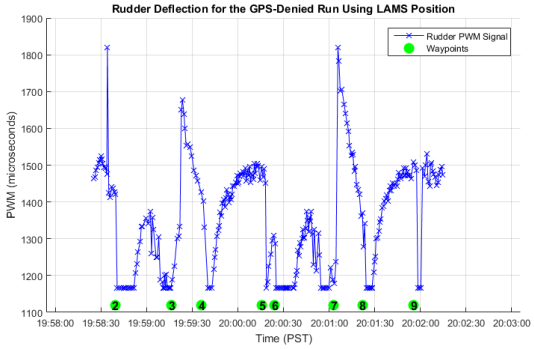
(b) Crosstrack error for the Auto run

Figure 19. Crosstrack error for the two control runs using GPS.

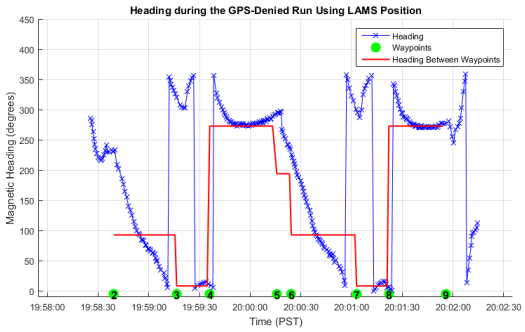
heading being commanded still more inaccurate. As the aircraft neared the next waypoint, the packets it received began to trigger much more extreme turns, as the aircraft suddenly realized it was much closer to the waypoint than it had previously thought. This is what caused the circling behavior seen in the vicinity of waypoints 3, 4, 7, and 8.

Figures 20(a) and 21(a) show the PWM signal commanding the rudder deflection for the WSTR runs using LAMS and ADS-B position data respectively. A centered, or neutral rudder corresponds to a PWM of 1500 microseconds. Full leftward deflection of the rudder corresponds to a PWM of 1000 microseconds, and full rightward deflection corresponds to 2000 microseconds. In order to prevent the firmware from over-commanding the rudder and potentially inducing an unsafe flight condition, the rudder deflection was intentionally limited to $\pm 30°$, which corresponds to a slightly decreased PWM range that varies from aircraft to aircraft. The lower limit for the aircraft used in the final demonstration was 1167 microseconds, meaning that the plateaus in the rudder deflection at this value correspond to full leftward deflection of the rudder. Figures 20(b) and 21(b) show the magnetic heading of the aircraft measured by the primary compass for the two respective WSTR runs. Note that the sharp spikes on these plots do not necessarily indicate a large change in heading, merely that the plane was traveling almost exactly due north, causing the heading to fluctuate between $0°$ and $360°$.

Examining Figures 20(a) and 21(a), we see that the rudder deflection between waypoints 2 and 3, and

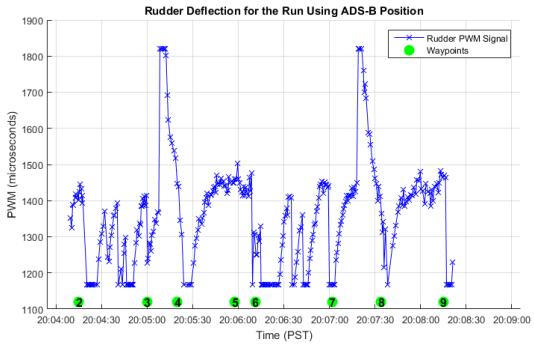


(a) The PWM signal being sent to the rudder.

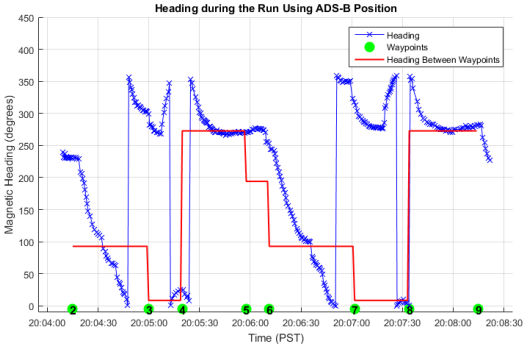


(b) The magnetic heading of the aircraft.

Figure 20. Steering controller behavior during the GPS-denied LAMS run.



(a) The PWM signal being sent to the rudder.



(b) The magnetic heading of the aircraft.

Figure 21. Steering controller behavior during the ADS-B run.

between 6 and 7, is not as extreme as would be expected for an aircraft as far off course as the plane actually was. Ideally, during these portions of the flight path, the aircraft should have maintained a constant heading of approximately 85°. In actuality, as seen in Figures 20(b) and 21(b), the heading changed slowly from around 200°, and was forced to overshoot and circle back around to a heading of around 300° to reach the waypoint. For much of this time, the rudder was receiving a PWM signal of between 1200-1400 microseconds, which is far too gentle of a turn for an aircraft that far off course. This gentle turn resulted in a slow arc between waypoints, rather than a crisp turn followed by a straight line. This gentle turn was a direct result of the lag in the transmission of position data from the ground station, as discussed above.

V. Conclusions and Further Research

This paper demonstrates an architecture to allow a UAS to operate in GPS-denied environments. The main innovation is a combination of hardware and software that allows a UAS equipped with a standard aviation transponder to be tracked by a ground-based multilateration system. Flight tests were performed to demonstrate the viability of the system. Using TRAPIS, a UAS was successful in navigating through a set of waypoints without using GPS while operating within range of the ground-based multilateration system.

The only significant deviations from the planned trajectory occurred due to a combination of high winds and a perceived lag in the receipt of position data by the UAS. The aircraft's handling of challenging wind conditions could be improved by the implementation of a more complex flight mode allowing the use of ailerons for steering in addition to the rudder. A more complex controller that incorporates derivative or integral control would also allow the steering controller to be tuned more aggressively than the pure proportional controller used here. This would decrease the response time of the steering controller and increase its reliability in remaining on a designated flight path. The lag in data transmission is most likely

due to the fact that the aircraft's position was being computed on the ground station rather than on the aircraft itself. The system of processing navigation data on the ground and then routing it back up to the aircraft introduces an inherent time delay. This could be alleviated by introducing a mechanism that allows for direct communication between the aircraft and the LAMS unit, without the need to route information through the ground station as well, or by simply increasing the efficiency of the hardware and software systems associated with the ground station.

Future development based upon this project will involve the use of new sources of position information. AFSL plans to investigate the use of cell phone towers as a ground-based source of position data in GPS-denied environments. This would allow a UAS to navigate autonomously anywhere where there is cellular reception, vastly expanding the area in which UAS can fly safely.

Acknowledgments

The authors would like to thank Andy von Flotow from Hood Technology for sponsorship of the research, guidance, and contributions to the research vision.

Advanced Navigation and Positioning Corporation (ANPC) was instrumental during the progression of the project. Michael Van Dooren provided invaluable technical support and assistance navigating ANPC software. Karl Winner provided technical support and assistance interfacing UW software with the ANPC LAMS during flight testing. Jeff Mains also provided support of the project.

Sagetech Corporation, particularly Kelvin Scribner, James Davis, and Tom Furey, contributed technical expertise and equipment.

Rolf Rysdyk from Insitu assisted with guidance and flight testing support.

Many members of the UW's Autonomous Flight Systems Laboratory contributed to the research including Taehan Kook, Connor Kafka, and Selina Lui.

Carnation Farms generously supported the flight testing of this project by allowing tests to be conducted at their facility.

Brian Prange and Jeff Renard and the Columbia Gorge Regional Airport helped coordinate airspace and usage of KDLS for the final flight test.

Finally, the authors sincerely thank Juris Vagners for supporting this work and other activities within the Autonomous Flight Systems Laboratory.

This research was sponsored by the Joint Center for Aerospace Technology Innovation (JCATI) as part of the project entitled "UAV Operations in GPS-Denied Skies."

References

[1]United States Superintendent of Documents, "FAA Modernization and Reform Act of 2012," Tech. rep., Federal Aviation Administration, 2012.

[2]"FAA Makes Progress with UAS Integration," 2006, `http://www.faa.gov/news/updates/?newsId=68004`.

[3]"FAA's NextGen Implementation Plan," Federal Aviation Administration.

[4]Federal Aviation Administration, "NextGen UAS Research, Development and Demonstration Roadmap," Tech. rep., Federal Aviation Administration, 2012.

[5]Scovel, C. L., "FAA's Implementation of the FAA Modernization and Reform Act of 2012 Remains Incomplete," Tech. rep., Department of Transportation Inspector General, 2014.

[6]Association for Unmanned Vehicle Systems International, "The Economic Impact of Unmanned Aircraft Systems Integration in the United States," Tech. rep., Association for Unmanned Vehicle Systems International, 2013.

[7]Lum, C. W. and Vagners, J., "A Modular Algorithm for Exhaustive Map Searching Using Occupancy Based Maps," *Proceedings of the 2009 Infotech@Aerospace Conference*, Seattle, WA, April 2009.

[8]Lum, C. W., Vagners, J., and Rysdyk, R. T., "Search Algorithm for Teams of Heterogeneous Agents with Coverage Guarantees," *AIAA Journal of Aerospace Computing, Information, and Communication*, Vol. 7, January 2010, pp. 1–31.

[9]Lum, C. W., Gardner, S., Jordan, C., and Dunbabin, M., "Expanding Diversity in STEM: Developing International Education and Research Partnerships in a Global Society," *Proceedings of the American Society for Engineering Education 123rd Annual Conference & Exposition*, June 2016.

[10]Lum, C. W., Mackenzie, M., Shaw-Feather, C., Luker, E., and Dunbabin, M., "Multispectral Imaging and Elevation Mapping from an Unmanned Aerial System for Precision Agriculture Applications," *Proceedings of the 13th International Conference on Precision Agriculture*, St. Louis, MO, August 2016.

[11]Lum, C. W., Summers, A., Carpenter, B., Rodriguez, A., and Dunbabin, M., "Automatic Wildfire Detection and Simulation Using Optical Information from Unmanned Aerial Systems," *Proceedings of the 2015 SAE Aerotec Conference*, Seattle, WA, September 2015.

[12]Lum, C. W., Rysdyk, R. T., and Pongpunwattana, A. A., "Autonomous Airborne Geomagnetic Surveying and Target Identification," *Proceedings of the 2005 Infotech@Aerospace Conference*, AIAA, Arlington, VA, September 2005.

[13]Ueunten, K. K., Lum, C. W., Creigh, A. A., and Tsujita, K., "Conservative Algorithms for Automated Collision Awareness for Multiple Unmanned Aerial Systems," *Proceedings of the 2015 IEEE Aerospace Conference*, Big Sky, MO, March 2015.

[14]Lum, C. W., Rowland, M. L., and Rysdyk, R. T., "Human-in-the-Loop Distributed Simulation and Validation of Strategic Autonomous Algorithms," *Proceedings of the 2008 Aerodynamic Measurement Technology and Ground Testing Conference*, Seattle, WA, June 2008.

[15]Lum, C. W., Vagners, J., Jang, J. S., and Vian, J., "Partitioned Searching and Deconfliction: Analysis and Flight Tests," *Proceedings of the 2010 American Control Conference*, Baltimore, MD, June 2010.

[16]Lum, C. W., Vagners, J., Vavrina, M., and Vian, J., "Formation Flight of Swarms of Autonomous Vehicles in Obstructed Environments Using Vector Field Navigation," *Proceedings of the 2012 International Conference on Unmanned Aircraft Systems*, Philadelphia, PA, June 2012.

[17]Martel, F., Shultz, R., W, S., Wang, Z., and Czarnomski, M., "Unmanned Aircraft Systems Sense and Avoid Avionics Utilizing ADS-B Transceiver," *AIAA Aerospace Conference*, 2009.

[18]Lai, C., Ren, Y., and Lin, C., "ADS-B Based Collision Avoidance Radar for Unmanned Aerial Vehicles," *Microwave Symposium Digest*, 2009.

[19]Stark, B., Stevenson, B., and Chen, Y., "ADS-B for Small Unmanned Aerial Systems: Case Study and Regulatory Practices," *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2013.

[20]Ramasamy, S., Sabatini, R., and Gardi, A., "Avionics Sensor Fusion for Small Unmanned Aircraft Sense-and-Avoid," *Meteorology for Aerospace*, 2014.

[21]Lum, C. W., Larson, R. S., Handley, W., Lui, S., and Caratao, Z., "Flight Testing an ADS-B Equipped sUAS in GPS-Denied Environments," *Proceedings of the AIAA Flight Testing Conference*, Denver, CO, June 2017.

[22]Larson, R. S., Winde, J., and Lum, C. W., "UAS Position Estimation in GPS-Degraded and Denied Environments Via ADS-B and Multilateration Fusion," *Proceedings to the AIAA Information Systems-AIAA Infotech@ Aerospace Conference*, Kissimmee, FL, January 2018.

[23]Larson, R. S., *sUAS Position Estimation and Fusion in GPS-Degraded and GPS-Denied Environments using an ADS-B Transponder and Local Area Multilateration*, Master's thesis, University of Washington, Seattle, WA, March 2017.

[24]Handley, W., *Two NextGen Air Safety Tools: An ADS-B Equipped UAV and a Wake Turbulence Estimator*, Master's thesis, University of Washington, Seattle, WA, June 2016.

[25]Advanced Navigation and Positioning Corporation, "TLS: Transponder Landing System," Tech. rep., Advanced Navigation and Positioning Corporation.

[26]Meier, L., Tanskanen, P., Heng, L., Lee, G., Fraundorfer, F., and Pollefeys, M., "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Autonomous Robot*, 2012.

[27]Lum, C. W. and Tsukada, D. A., "UAS Reliability and Risk Analysis," *Encyclopedia of Aerospace Engineering*, July 2016.

[28]Lum, C. W. and Waggoner, B., "A Risk Based Paradigm and Model for Unmanned Aerial Vehicles in the National Airspace," *Proceedings of the 2011 Infotech@Aerospace Conference*, St. Louis, MO, March 2011.

[29]Lum, C. W., Gauksheim, K., Kosel, T., and McGeer, T., "Assessing and Estimating Risk of Operating Unmanned Aerial Systems in Populated Areas," *Proceedings of the 2011 AIAA Aviation Technology, Integration, and Operations Conference*, Virginia Beach, VA, September 2011.