JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 5, April 2008

Networked Communication, Command, and Control of an Unmanned Aircraft System

Eric W. Frew* , Cory Dixon† , Jack Elston‡ , Brian Argrow§ , and Timothy X. Brown¶
Research and Engineering Center for Unmanned Vehicles
University of Colorado, Boulder, Colorado 80302

DOI: 10.2514/1.26558

Fully autonomous, cooperative, multivehicle operation requires the development and integration of various levels of intra- and intervehicle communication, sensing, control, and autonomy. This paper describes the Networked unmanned aircraft system Communication, Command, and Control (NetUASC3) architecture that combines meshed network intelligence, mission-level tasking information, and automatic flight control into an integrated unmanned aircraft system. The NetUASC3 architecture described includes: the University of Colorado at Boulder Ares unmanned aircraft, the onboard flight management architecture, and monitoring and command and control software that exploits the existing ad hoc Unmanned Aircraft System Ground network mesh network. Results from flight demonstration of the NetUASC3 architecture are provided to highlight the interplay between the various subsystems. Specific results demonstrate sensor-reactive and communication-reactive control, meshed network performance, atmospheric data collection, validation of radio-propagation models, and delivery of streaming video over a multihop airborne–ground network.

Nomenclature

$P_{i,j}$	the power of the jth emitter received at the location of the ith aircraft
P_j	received power of jth transmitter
P_0, d_0, k_0, e	parameters of empirical radio propagation model
$\boldsymbol{p}_j$	2-D position vector or jth radio transmitter
U	constant UA speed
$\boldsymbol{u}$	input vector for entire UA team
u_i	turn rate command of ith UA
(x, y, ψ)	2-D x-position, y-position, and heading of UA
$\boldsymbol{x}_i, \boldsymbol{x}$	vector state of ith UA, entire UA team
$\omega_{\max}$	maximum turn rate command

Received 14 July 2006; revision received 29 May 2007; accepted for publication 11 February 2008.

* Assistant Professor, Aerospace Engineering Sciences, eric.frew@colorado.edu, AIAA Member.
† Graduate Student, Aerospace Engineering Sciences, cory.dixon@colorado.edu, AIAA Member.
‡ Graduate Student, Aerospace Engineering Sciences, jack.elston@colorado.edu, AIAA Member.
§ Professor, Aerospace Engineering Sciences, brian.argrow@colorado.edu, Senior Member.
¶ Associate Professor, Interdisciplinary Telecommunications, timxb@colorado.edu, Non-Member.

I. Introduction

SMALL unmanned aircraft (UA) that weigh approximately 10 kg have proved useful in a variety of operations such as reconnaissance [1], atmospheric sensing [2–4], and biological sampling [5]. Cooperative multi-UA operations have the potential to expand the useful envelope to cooperative search and localization, volumetric atmospheric sampling, and airborne networking to support ground communications. Multi-UA operations are expensive if a separate pilot is required for each vehicle. To reduce costs, operator control can be simplified to higher-level tasks for a group of UA as a whole. UA would autonomously cooperate to complete the assigned group tasks.

Some initial experiments have considered multivehicle operations with small UA [6–10]. Others have considered different control algorithms for formation flying [11–13], simple cooperative rules [14–16], or centralized control of multiple vehicles [17] mainly in simulated environments. While these treat different aspects of the problem, they do not address the complete picture for realizing a fully autonomous cooperative multivehicle unmanned aircraft system (UAS).

Fully autonomous, cooperative, multivehicle operation requires the development and integration of various levels of intra- and intervehicle communication, sensing, control, and autonomy. The work presented in the current paper is part of a multistage effort (Fig. 1) to create a multivehicle UAS. These stages build on earlier work on a wireless network of UAS and ground nodes denoted as an ad hoc UAS Ground Network (AUGNet) [18, 19]. AUGNet (described in Sec. II. A) provides the basic wireless ad hoc networking protocols and hardware used in the NetUASC3 architecture.

In stage 1, an intelligent avionics system is designed and implemented to enable single-vehicle autonomy. Subsystem development and intra-UA communication and control are key elements to this stage. Additional elements of the stage 1 system include: command and control of a single UAS through a fully meshed mobile ad hoc network; operator interface design to facilitate remote management of the UAS, sensor payloads, and the meshed network; and sensor payload integration.

In stage 2, the intelligent control capabilities enabled by stage 1 will be used by a single UA to demonstrate semi-autonomous flight. Onboard flight management and intelligence will allow the UAS to perform various tasks based on mission-level objectives; local sensor data collected in flight; and operator commands sent through the meshed ad hoc network. Example applications to be demonstrated in stage 2 include aircraft leashing (also called electronic tethering) in which a single UA tracks a possibly moving ground node to provide a communication relay from that node to other nodes in the environment [20]. The inter-UA communication capability of the AUGNet system will be expanded in stage 2 to enable multi-UA control.

Finally, stage 3 will demonstrate collaborative multi-UA decision making and control. The AUGNet system will be exploited fully to allow a single operator to task a flock of UA through a meshed network in which only a subset of UA have a direct connection (i.e., line-of-sight) to the operator control station. Commands will flow outward from

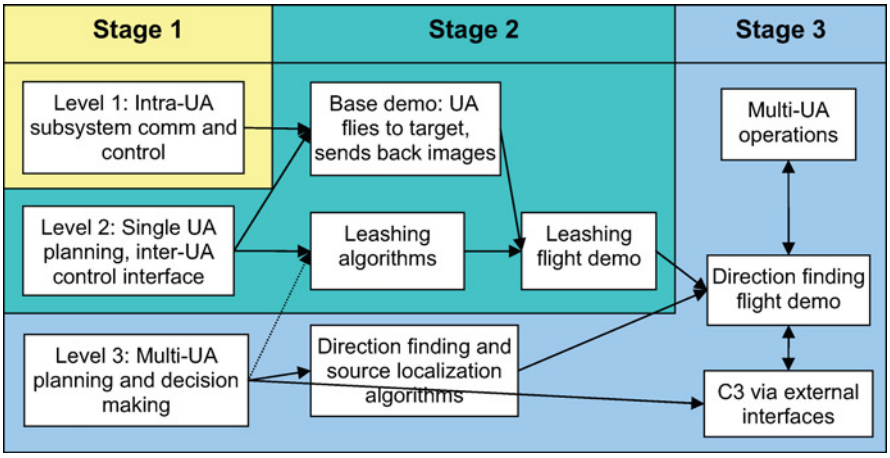


Fig. 1 Staged approach to Networked C3 for UA team.

multiple dispersed operators/users to the UA flock, which will deliberate among its members on how best to achieve the commanded objectives. At the same time, sensor and network data will flow back through the ad hoc network in a fault-tolerant manner such that data are delivered to the proper users even though end-to-end connection of a link may never be possible. Radio source localization [21] and cooperative search, acquisition, and tracking (CSAT) [22] are two example applications that could be demonstrated in stage 3.

The present paper describes the stage 1 design of an intelligent aerial platform and a meshed, mobile ad hoc network denoted Networked UAS Communication, Command, and Control (NetUASC3). Control systems that allow for autonomous networked operation are integrated into existing UAS platforms. Software algorithms tie sensor measurements, network intelligence (i.e., RF signal level, data throughput, etc.) and mission-level tasking information into automatic flight controls. The resulting flight management architecture refines UAS control systems and software; advances UAS communication, command and control technology; and provides a hardware base for a multi-UA network for missionizing group-UAS applications. Within this larger context, this effort focuses on small UA ($\sim$10 kg) because they are small enough to allow cost-effective development and testing of multiplane control algorithms while being large enough to carry interesting networking and sensor payloads.

The remainder of this paper is organized as follows. Section II describes the main components of the NetUASC3 system, with an emphasis on the components that extend the capability of the existing AUGNet system. A modular avionics system is described that enables intelligent flight management of various onboard subsystems including low-level flight control meshed communication, payload sensors, and higher-level supervisory control. Section III then describes results from flight demonstrations of the NetUASC3 architecture that highlight the interplay between the various subsystems. Specific results demonstrate sensor-reactive and communication-reactive control, meshed network performance, atmospheric data collection, validation of radio-propagation models, and delivery of streaming video over a multihop airborne–ground network.

II. NetUASC3 Architecture

The NetUASC3 architecture integrates autonomous FC, network communication, and sensor subsystems to form an intelligent information-gathering UA platform. New software and hardware components are combined with custom and commercial off-the-shelf (COTS) components developed as part of the AUGNet system [18, 19]. The main additions in the stage 1 architecture presented here are the onboard flight management architecture, refinement of the virtual cockpit (VC) interface for semi-autonomous control over the mesh network, and integration of several new sensor payloads. A hardware-in-the-loop (HIL) laboratory test bed was also constructed so that hardware, software, and interoperability elements could be rapidly developed and tested.

Figure 2 shows a schematic of the NetUASC3 architecture. Networked command and control is enabled by combining the AUGNet mesh network with an onboard FMS. The AUGNet allows meshed communication between the UAS and operators and network monitoring stations located in the field or remote operators located on the internet. Command information is sent to the vehicle while telemetry and sensor data are back-hauled to the operator in addition to a database located at the University of Colorado. The onboard FMS comprises several subsystems connected through custom interface boards shown in the bottom center of Fig. 2. These subsystems include: the PiccoloPlus autopilot, which measures position and orientation information, operates the UA control surfaces, and guides the UA to waypoint or flight pattern goals; the communication module (Soekris Board and Soekris Interface Node), which interfaces between the AUGNet and the onboard system; the sensor payload subsystems, which can connect directly to the meshed AUGNet; and the supervisory flight computer that manages the higher-level functionalities of the aircraft by interpreting commands and making mission-level decisions. The remainder of this section describes the components of the NetUASC3 architecture in more detail.

A. AUGNet

The AUGNet system is a wireless multi-hop network based on COTS IEEE 802.11b (WiFi) hardware that combines static ground nodes, moving ground nodes, and aerial nodes carried by autonomous UAs connected through a mesh network back-hauled to the internet (Fig. 3). The main components of the AUGNet system [23, 24] include: custom-made mesh network radios; network monitoring software and a database that can be accessed during flight experiments from the internet; routing software using the click modular router to implement dynamic source routing (DSR); and the CU Ares aircraft (see Sec. II. F) designed and manufactured at The University of Colorado (CU) Boulder.

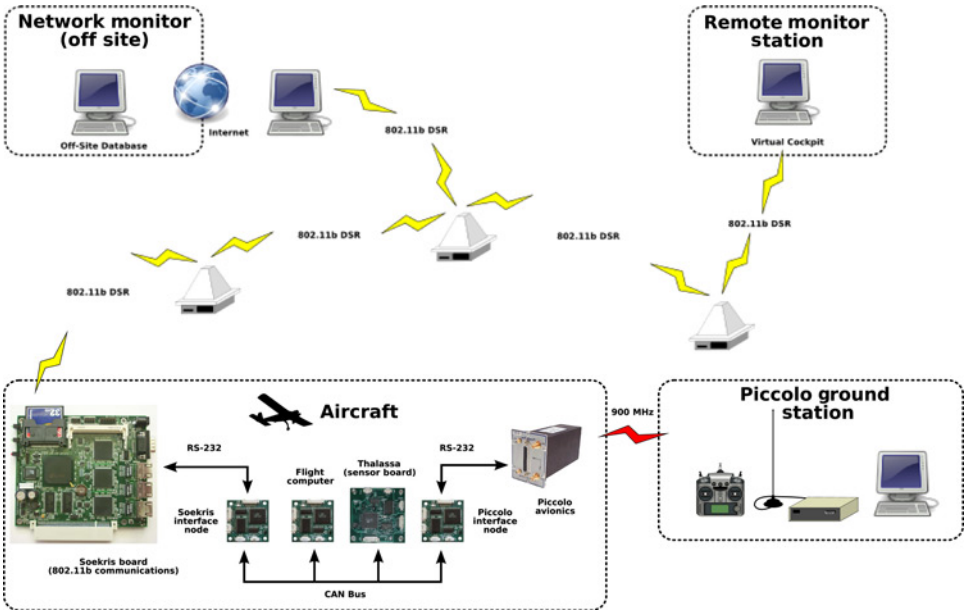


Fig. 2 Stage 1 NetUASC3 architecture.

The mesh network radio (MNR) consists of an Atheros AR5213A chipset mini PCI card, a Soekris 4511 single board computer, a Fidelity-Comtech 1-W bidirectional amplifier, and a Garmin global positioning system (GPS). The radio runs the DSR ad hoc routing protocol that enables the aircraft to communicate with other aircraft and with similar ground nodes. The DSR protocols developed under this project are available as an open source download at http://pecolab.colorado.edu. A packet communication scheme [23, 24] is used to communicate data objects and commands reliably between network nodes, including the VC (see Sec. II. G) and Ares, over the wireless link.

B. Onboard Flight Management Architecture

Onboard flight management is achieved through a modular embedded control architecture that integrates custom and COTS components with onboard network interfaces. Modularity and module independence are achieved by maintaining intelligent system nodes, and providing bus protocols that guarantee message transmission. The FMS

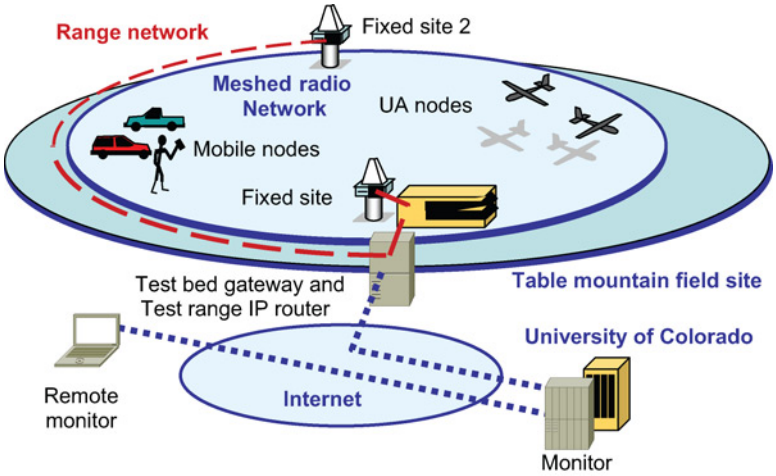


Fig. 3 AUGNet test bed.

consists of the communication subsystem, the autonomous flight control module, the computational module, and a collection of sensors in the sensor modules subsystem. The FMS is monitored and controlled from remote monitor stations (RMS) using the VC through AUGNet. Figure 4 is a schematic of the modular architecture showing the major subsystems and their connections.

For subsystems that have a built-in internet protocol (IP) interface, such as a network camera, either a serial line IP protocol (SLIP) or ethernet can be used to connect directly to the communication subsystem. Using network address translation (NAT) protocols and routing software on the communications interface (Soekris board) messages can be passed to onboard devices directly (see Fig. 5). These devices must implement at least a minimal set of the transmission control protocol (TCP/IP) functionality, and can include hardwired ethernet devices such as network cameras, or serial devices using SLIP.

The communication between onboard subsystems and the external AUGNet is handled using a form of NAT known as port forwarding. Port forwarding (sometimes referred to as tunneling) is the act of forwarding a network socket connection on a specific port from one network node to another IP-addressed node and port, thus providing the mechanism for an external network client to reach a port on a private IP address (inside a local area network) from the outside. This allows remote clients to address individual subsystems or sensors so that direct communication between any two subsystems within AUGNet or on individual UA is possible. For example, the VC (Sec. II. G) can send a command directly to a network camera onboard a specific Ares to pan in a certain direction or to download an image. Figure 5 provides a schematic overview of the NAT implementation on the Ares UA and shows the port assignments that are currently used. To make a connection to the IP-based network camera, a connection to the Ares UA is made on port 80. Within the onboard communication subsystem, port 80 is forwarded directly to the ethernet connection between the camera and the Soekris board. To connect to the PiccoloPlus autopilot, a connection on port 1000 is made to the aircraft. In this case, the Soekris communication board has port 1000 mapped to the SLIP interface to the communication interface node (the Naiad). This Naiad further translates the IP packet to the controller area network (CAN) protocol and the message is sent to the autopilot.

Most of the onboard subsystems cannot implement a minimal TCP/IP stack and are connected through a controller area network bus. For this subnet, custom-made interface boards (so-called Naiad boards [25]) are used as intelligent interpreters between the subsystems and the CAN bus. These nodes are distributed throughout the aircraft to connect the supervisory flight management computer, the autonomous flight system, ad hoc communication hardware, and various other sensors using the CAN bus. Each Naiad node maintains responsibility for the interface to a particular

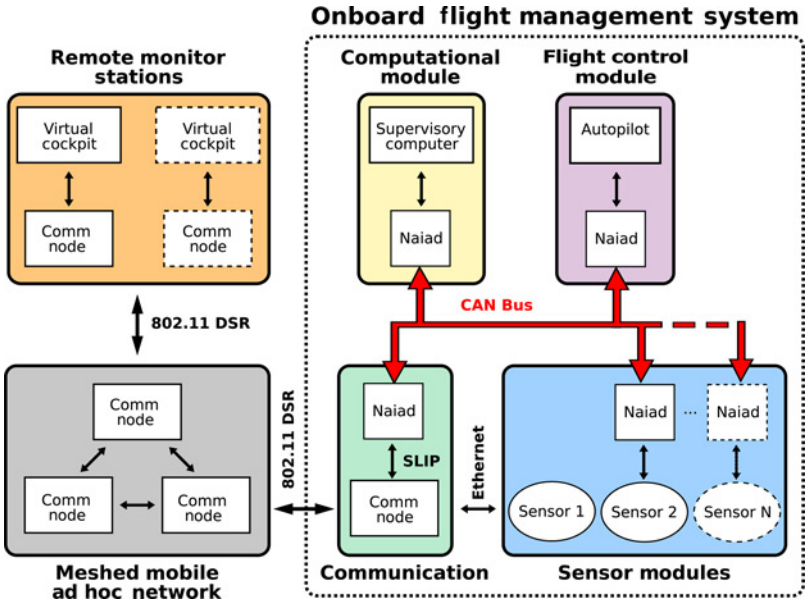


Fig. 4 UA onboard flight management architecture.

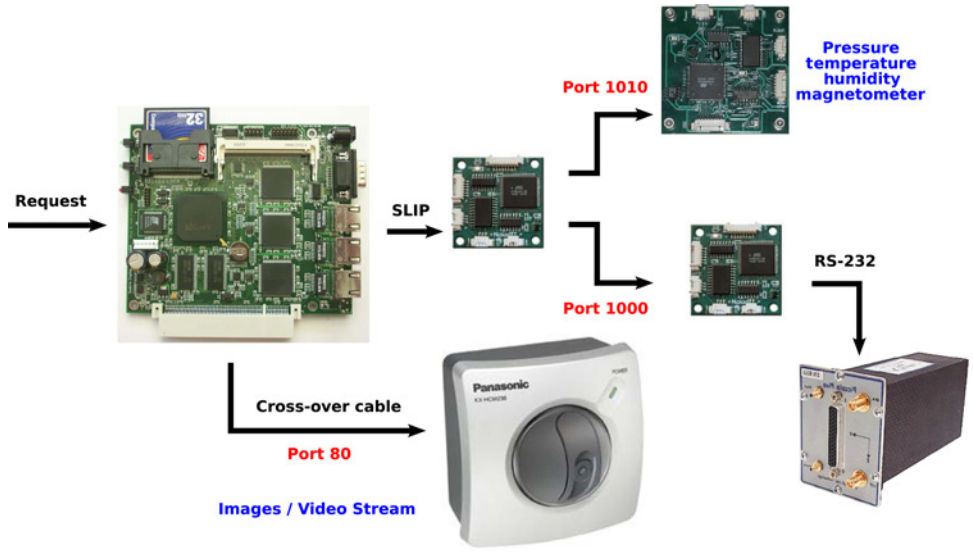


Fig. 5 Network address translation is used to direct communication between AUGNet to individual subsystems on the UA.

subsystem or a given computational task. With this architecture, subsystems can be replicated several times across the bus and allowed to broadcast simultaneously or in a controlled manner using local decision making or control by a supervisory health and status subsystem. Computationally intensive tasks such as filtering can be abstracted to a set of nodes tasked with data processing.

Flight management is currently performed by a single Naiad node, which has sufficient resources for the experiments presented in Sec. III. Other, more capable, processing platforms can be integrated to meet future needs through the Naiad RS232 interface using SLIP. The flight control system is the Cloud Cap Technology PiccoloPlus autopilot. The communication hardware is an AUGNet MNR described above. The MNR interfaces with the Naiad using the SLIP protocol over a RS232 serial port running at 57600 baud. The sensor modules include the Thalassa weather sensor package developed in-house and an IP-based network camera (Panasonic Model BB-HCM331A). The Naiad, Thalassa, PiccoloPlus, and supervisory controller are described in Secs. II. C and II. D.

C. Naiad and Thalassa Interface Nodes

The Naiad** interface node was originally developed as the core processor for a distributed avionics system [25]. In the NetUASC3 system Naiads connect individual components through the fault-tolerant, high-speed CAN serial bus. Each interface board supports a single subsystem (sensor module, communications, and so on) and serves as an intelligent interpreter between different subsystem components. The Naiad system allows the interface board to handle all low-level tasks such as polling instrumentation, and allows the bus protocol to remain high-level. In addition, the central flight computer can be responsible for system management through connections to the interface boards via the CAN bus. Figure 6a is a photograph of two Naiad nodes to show the board population and the connector layout.

The Naiad nodes are based on the Atmel ATMega128 microcontrollers, which provide pulse-width modulated (PWM) output for servo control, an analog-to-digital converter for sensor signal acquisition, and various other buses (I2C, SPI, UART). A system diagram for the node is shown in Fig. 7. A Naiad node is approximately 3.8 cm on each side, weighs 14.0 g, and nominally uses 50 mA at 5 V. The node contains a watch-dog timer with brown-out detection, and with the 14-MHz clock the microcontroller is capable of approximately 14 MIPS throughput. The node was constructed using a 4-layer board to minimize onboard signal noise and decrease footprint. Each has mounting holes

**Naiad and Thalassa are Neptune's first and second moons, respectively.

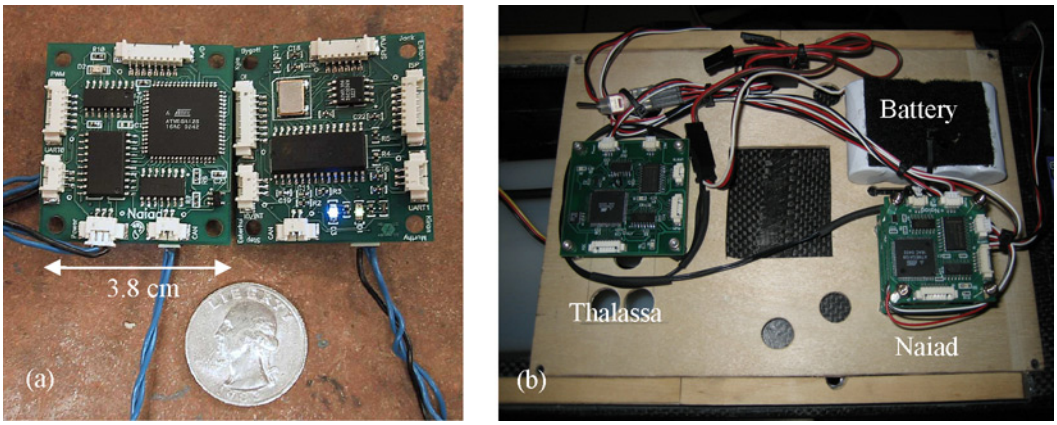


Fig. 6 (a) Top and bottom view of two Naiad boards. (b) Naiad and Thalassa boards mounted with battery power supply before installation.

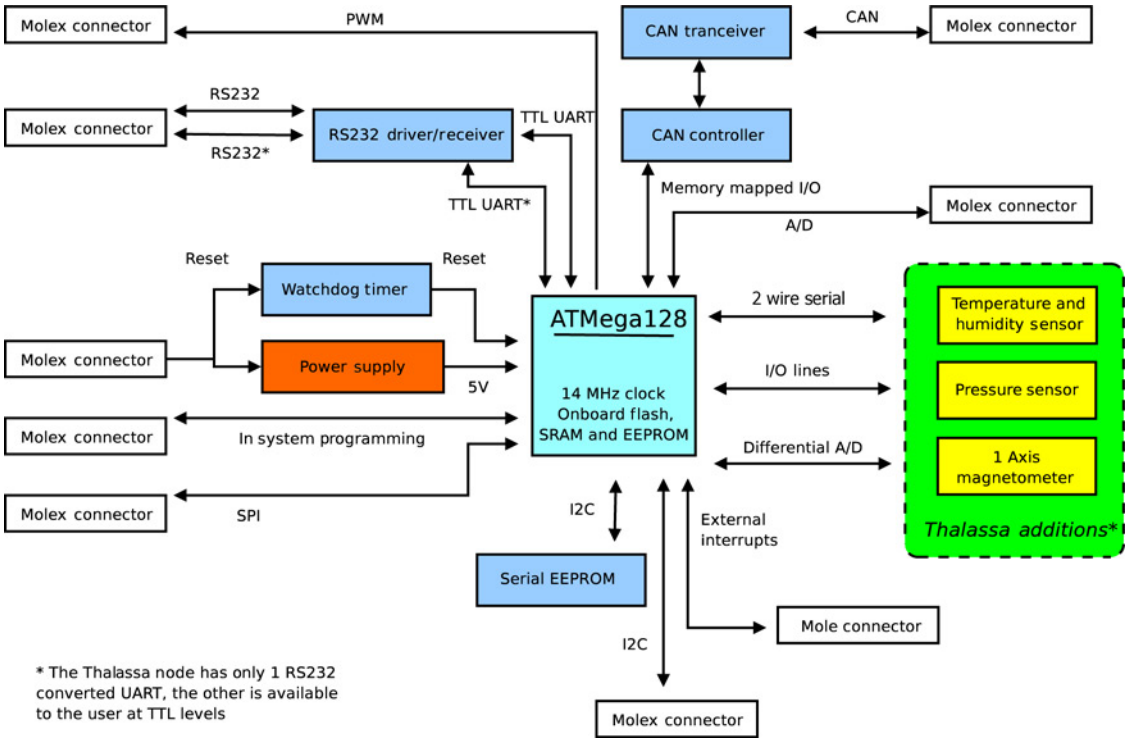


Fig. 7 Naiad and Thalassa block diagram.

to enable easy placement about the aircraft, or to be stacked in a convenient location with small standoffs. The Naiad node uses the SJA1000 CAN controller and TJA1053 fault-tolerant transceiver for inter-node communication.

The CAN bus supports data exchange rates up to 1 Mbps. Unlike other protocols, where each node reads messages to its address only, CAN allows a node to subscribe to and broadcast message "types." For example, a temperature measurement might be broadcast onto the bus as a type 'temperature' and a pressure measurement might be a type "pressure." If the central computer node is subscribed to these message types, it will accept a temperature or pressure message without the need for the sender node to specify the central computer's address. This enables several nodes

to broadcast a certain message and to be switched in and out of the system without having to reprogram the other interface nodes.

Thalassa is an expanded Naiad node with miniature temperature, pressure, and humidity sensors integrated onto the circuit board. A magnetometer is also integrated onto the node, but is not used for the current application. The Naiad was used as a base system for communications across the CAN bus, sensor data acquisition, and the use of the Naiad firmware. This resulted in a significant savings in both development time and cost. The software on the Thalassa board provides the high-level interface for all three sensors. Naiad and Thalassa nodes are shown with a battery pack in Fig. 6b. This is the mounting configuration for flight experiments. Figure 7 is a block diagram of the Naiad and Thalassa circuit boards. The diagram emphasizes that the circuitry is the same for both boards with the addition of the Thalassa sensors enclosed in the green box.

D. Autopilot and Supervisory Computer

The COTS PiccoloPlus autopilot is used for low-level control in the NetUASC3 architecture. The PiccoloPlus is the second generation of the Piccolo autopilot system developed by Cloud Cap Technology [26]. It is a complete integrated avionics system including the core autopilot, flight sensors, navigation, wireless point-to-point communication, and payload interface capabilities all in a compact package. The PiccoloPlus provides three modes of UA operation: manual pilot control, GPS waypoint navigation, and steering control. Manual pilot control is used for takeoffs and landings of the Ares aircraft while GPS waypoint navigation is the primary operation mode. In this mode, the autopilot system follows a series of GPS waypoints while holding altitude and airspeed. In steering control mode, the autopilot system maintains altitude and airspeed while following an externally provided turn (heading) rate, enabling the aircraft to be steered.

The complete PiccoloPlus system includes a ground station connected to a laptop PC and a manual resistance–capacitance (RC)-style pilot console. With the exception of the inertial measurement unit (IMU), the ground station contains exactly the same avionics package mounted in the UA. The PC provides the platform for the PiccoloPlus graphical operator interface (OI) that allows the user to track the UA GPS location, to monitor its performance, and to upload GPS waypoints or enter steering commands. Communication between the Cloud Cap ground station and the PiccoloPlus autopilot occur over a point-to-point 900 MHz link. Within the NetUASC3 system, this link is only used for manual operations and as a safety backup. All other communication occurs through the AUGNet mobile ad hoc network.

All commands that are exchanged between the ground station and the PiccoloPlus, including steering commands, can also be uploaded through an external RS232 serial interface on the PiccoloPlus. The Naiad in the automatic frequency control (AFC) uses this interface to communicate with the PiccoloPlus to obtain the health and status information, and to issue steering or GPS waypoint commands to the PiccoloPlus. The PiccoloPlus is capable of duplex operation so that commands from an onboard system can be intermixed with commands received over the 900 MHz link from a remote operator. Thus, control of the PiccoloPlus by a remote operator and onboard systems are not mutually exclusive modes.

The supervisory computer module acts as a central computer to the distributed system, running high-level mission control algorithms. The purpose of this module is to collect all of the data and telemetry from the other subsystems and use this information, along with mission goals and limits, to monitor system health, turn sensors on and off, and to provide sensor-reactive control commands to the autopilot system. These sensor-reactive control algorithms are also responsible for tying network intelligence and mission tasking into autonomous flight control commands.

For the flight tests described in Sec. III, the processing required to execute the experiment test plan was minimal so the supervisory computer algorithms were run directly on the Naiad interface node, reducing the AFC to a single component. If more processing is required in the future the Naiad can be interfaced with other computers as demonstrated in the communication module, or be extended with more Naiad*s* running specialized tasks within the computer module.

E. Payload Sensor

The Naiad interface system allows for a heterogeneous suite of sensor modules, independent of the sensor interface. This capability enables the notion of "sensor" to include traditional temperature sensors as well as the use of the

MNR as a sensor, measuring network performance metrics such as network connectivity, link throughput, and link signal strength. For the experiments described in Sec. III, the MNR metric of interest is connectivity with the VC.

To emulate a third-party scientific payload, the Thalassa sensor board was developed to provide temperature, pressure, and humidity measurements. Temperature and humidity are measured on a single chip manufactured by Sensirion AG, part number SHT15. The manufacturer reports a temperature resolution of 0.01 °C with accuracy of ±0.3 °C at 25 °C. The relative humidity is resolved at 0.03% with accuracy of ±0.3%. Absolute pressure is measured with the Intersema MS5534B Barometer Module. This module measures pressure in the 10–1100 mbar absolute pressure range, with a reported resolution of 0.1 mbar and an accuracy of ±1.5 mbar at 25 °C. As the Thalassa sensor payload was developed to test sensor-reactive control, and not necessarily provide high-quality atmospheric data, the accuracy and resolution reported by the manufacturer was not verified.

F. CU Ares UA

Flight demonstrations of the NetUASC3 architecture are carried out on the Ares aircraft (Fig. 8) constructed at CU Boulder. The airframe of the Ares aircraft is based on the layout of the Senior Telemaster, a popular RC model. Custom modifications include an expanded fuselage to accommodate a payload section and conversion from a tail-dragger configuration to one with tricycle gear. The monocoque fuselage, wings, main gear, and horizontal tail are made of a carbon composite laid over carbon-composite laminated plywood bulkheads and ribs. The Ares is

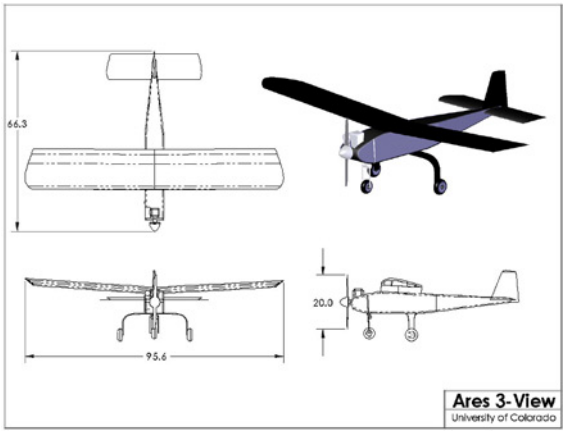


Fig. 8a Three-view and isometric view of the Ares UA, dimensions are in inches.

Fig. 8b Three Ares UAs on the ground at the local flight test facility.

powered by a 5-hp, two-stroke engine and has an additional payload capacity of 10 lbs. More details can be found in Dixon et al. [19].

G. Virtual Cockpit Remote Monitoring Station

The VCRMS was developed to enable UAS command and control over the 802.11 wireless network. The RMS consists of a Pentium level laptop and a Proxim ORiNOCO Gold 802.11a/b/g PCMCIA card with the AR5212 Atheros chipset. The laptop runs the Linux operating system and uses the same 802.11b DSR routing algorithm implemented on the MNRs to enable network communication. The laptop also runs the VC program, which provides the interface to the NetUASC3 system for a remote operator. In addition, the VC is responsible for logging the communication packets between the RMS and UAS to enable flight replay and post processing of the sensor data.

The VC is implemented in C++ using the OpenGL and GTK+ libraries. OpenGL and GTK+ are multi-platform toolkits for creating graphical user interfaces. Although the VC was primarily designed to run in a UNIX-based operating system, use of these libraries enables the VC to run in Windows as well, while providing the same look and functionality to the operator. The functional interface was chosen to follow, in general, the same scheme as provided by the PiccoloPlus OI. In most cases, the VC reproduces the same functional interface to the UAS as the OI. The main exception is manual piloting, which must go through the PiccoloPlus ground station. Thus, the goal of the RMS is not to replace the PiccoloPlus OI, but to extend the capabilities over a meshed network interface. Screen shots are shown in Fig. 9.

On the left of Fig. 9 is the main VC window, which provides the high-level situational awareness to the operator through an aerial perspective displaying ground station location (blue dot), GPS waypoint plans for a UA (green line and black dots), and UA location (red triangle) on a georeferenced aerial photograph. In addition to displaying GPS

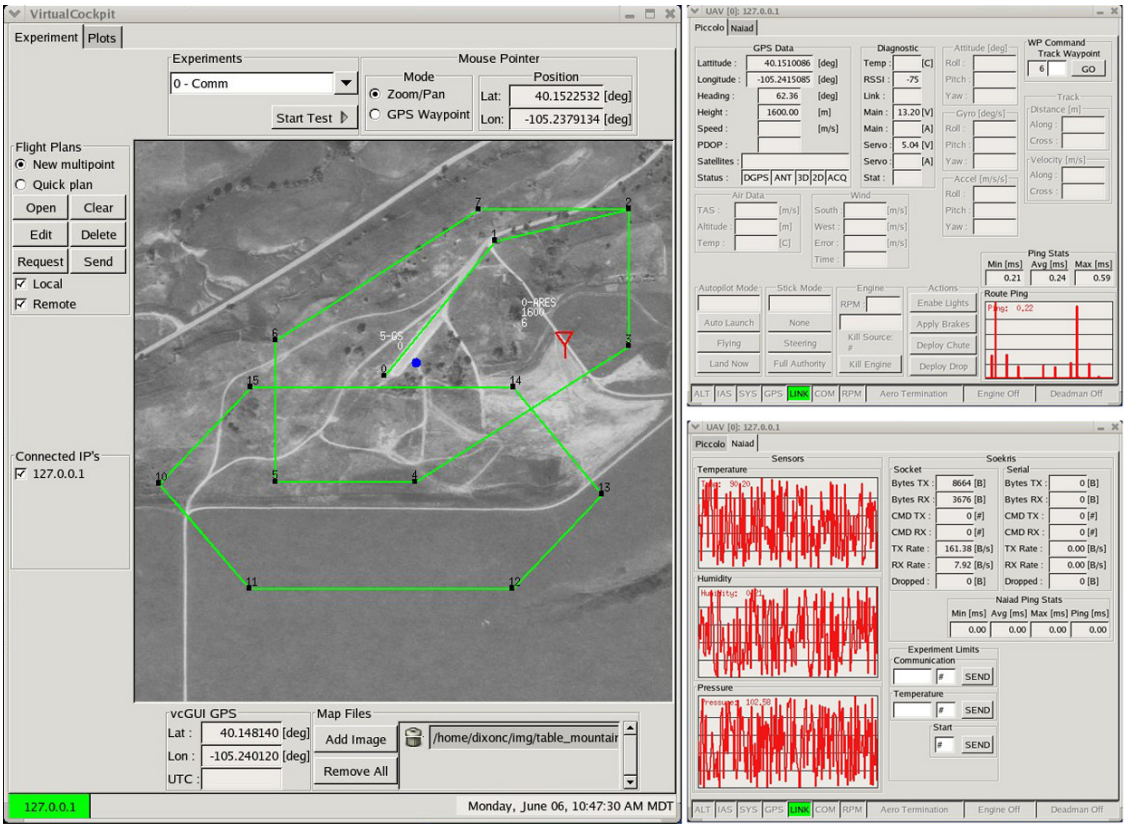


Fig. 9 Virtual Cockpit screen shots.

latitude and longitude of network nodes, the symbols are augmented with additional information to the upper left of the symbol: the node ID, altitude, and the next UA waypoint.

The two windows on the right are two panes provided to the operator for each UA that is connected on the network. The top pane provides PiccoloPlus health and status, as would be seen on the PiccoloPlus OI, in addition to displaying statistics from ping commands between the ground station and the UA. The bottom pane provides scrolling strip charts for the display of the Thalassa sensor data (temperature, humidity, and pressure) as well as displaying communication performance statistics measured onboard the UA for the network socket interface to the ground station and the serial interface to the Naiad communications node.

H. Hardware in the Loop Simulation

Development and testing of the NetUASC3 system were facilitated by hardware-in-the-loop simulation in a laboratory environment (Fig. 10). The laboratory test bed uses all of the hardware and software components in the full system, except for the actual sensor measurements of the PiccoloPlus unit. Instead, modified GPS, inertial, and air-data measurements are given as inputs to the navigation subsystem from flight simulator software provide by Cloud Cap Technologies. The navigation subsystem outputs the control surface commands to the flight computer.

The result is that the UA "thinks" it is flying while sitting on the lab bench, including deflection of the aerodynamic control surfaces and actuating the throttle. In this way, system interactions can be tested under simulated flight conditions, and the behavior of the UA as it switches from different regimes can be tested and debugged prior to an actual flight test. This enabled many software bugs and interoperability problems to be efficiently captured in a rapid-prototyping environment.

The laboratory environment is never completely realistic. During flights, links can be weak or vary while the UA maneuvers. These conditions are partially simulated by moving nodes around the lab building or shielding the radio antennas. During flights, the UA airframe and payloads are subject to engine vibration and other dynamic mechanical forces that are not simulated in the lab. The flight simulator has a vehicle model based on the Ares UA, but the model's flight characteristics can differ from what is observed during actual flights. Despite these deficiencies, the laboratory setup proved invaluable at completing the development and testing in a timely manner.

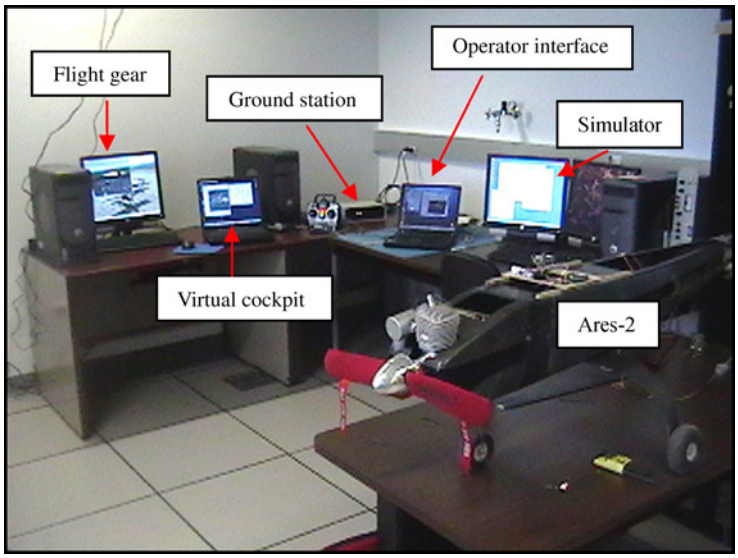


Fig. 10 Hardware-in-the-Loop testing in the laboratory.

III. Flight Demonstrations

A series of flight experiments to verify the NetUASC3 architecture were conducted at the Table Mountain Radio Quiet Zone near Boulder, Colorado. The layout of the airfield, flight plans, network operation center, mesh network radio nodes, and virtual cockpit are shown in Fig. 11. The test plan consisted of two experiments to test the ability of the aircraft to react to changes in sensor measurements or communication status.

A basic experiment consists of the following steps.

1) The Ares UA is manually piloted for takeoff and transitioned into flight plan 1.
2) Ares is commanded into autonomous mode and flies flight plan 1, which is preloaded.
3) A new flight plan is entered over the communication link followed by a "start experiment" command.
4) Ares transitions into flight plan 2 where it sends a sensor report every second consisting of temperature, pressure, and humidity data.
5) Ares maintains flight plan 2 until one of the following conditions is met.
   a. The temperature probe records a temperature below 40 °F (potential icing).
   b. The communication link between the RMS and UAV has been down for more than 40 s.
6) Ares transitions back into flight plan 1.
7) If another experiment is desired in the same flight, step 3 is repeated, otherwise Ares is manually landed.

The two experiments, based on this basic design, were a virtual icing experiment and a communication reactive flight experiment, each designed to trigger the different conditions in step 5. For the virtual icing experiment a temperature offset was subtracted from the temperature sensor reading. The offset starts at zero and decreases over

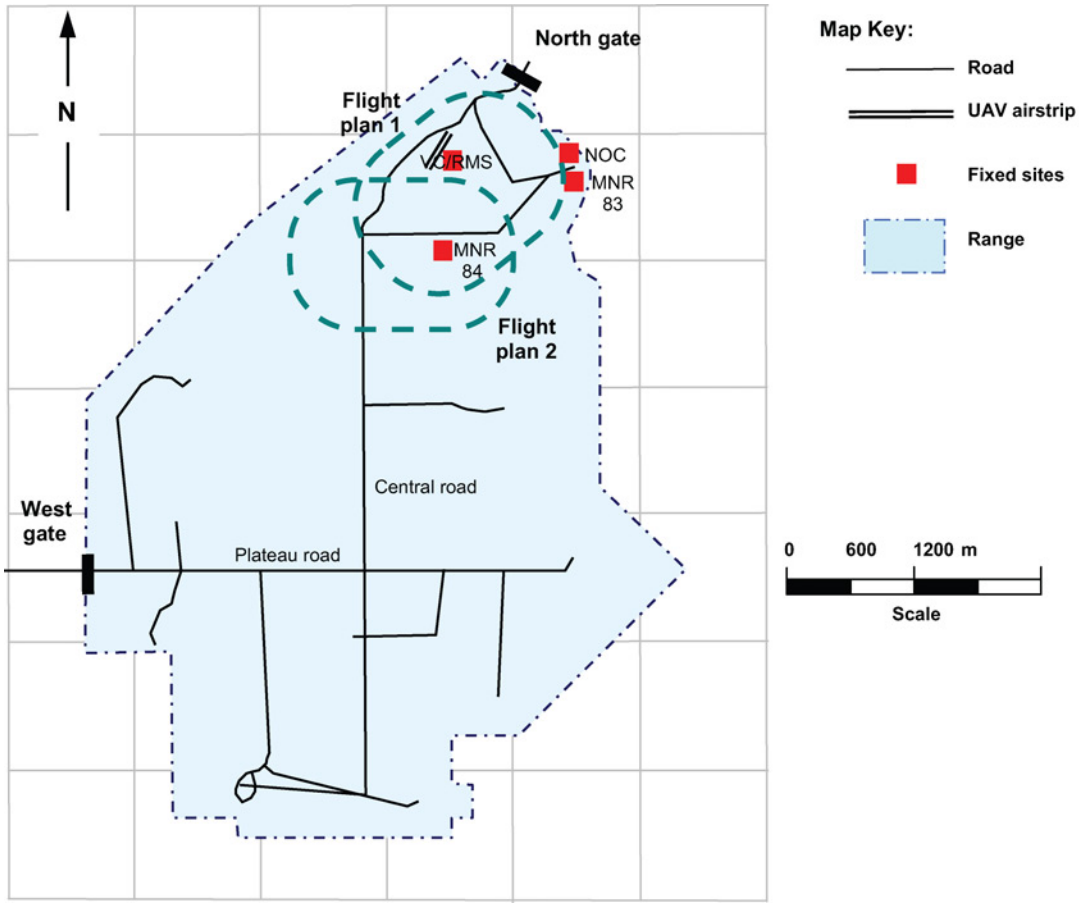
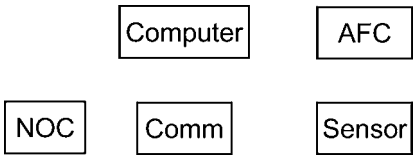
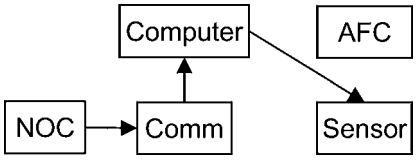
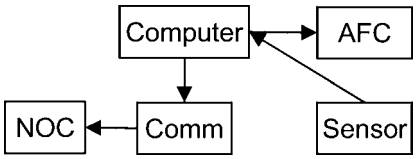
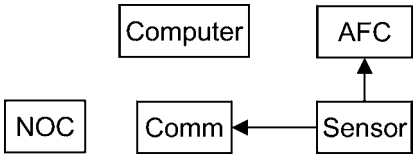
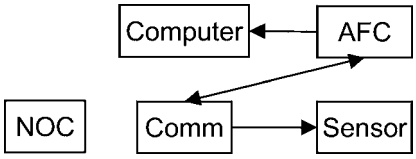


Fig. 11 Experimental setup at Table Mountain.

Table 1 Significance of steps in the experiments.

Step 1,2: Ares is launched and flies autonomously in preloaded flight plan 1.	
Step 3: A new flight plan is entered over the communication link along with a "start experiment" command which turns on the sensors.	
Step 4: Ares transitions into flight plan 2 where it sends a sensor report every second.	
Step 5a: Ares maintains flight plan 2 until the temperature probe records a temperature below 40 °F at which time the plane returns to flight plan 1 and sends a freeze warning.	
Step 5b: Ares maintains flight plan 2 until the communication to the RMS is down for more than 40 s at which time the plane returns to flight plan 1 and resets the sensor.	

time to simulate a dropping temperature. In the communication reactive flight experiment, the constant pinging between the UA and the ground station, that is used to measure network performance, is manually stopped to simulate a communication loss. By simply stopping the pings, which act as a "heartbeat" to both systems for link connectivity, sensor and UAS data can still be downlinked to the RMS.

The significance of the steps in these experiments is represented in Table 1. The arrows indicate which subsystem is influencing which other subsystem in each step. The table shows that every influence relationship (except from the AFC to the sensor) is tested.

A. Test Results

Experiment results were obtained in a 50-min flight of Ares-2, under full autonomous control for more than 30 min. The first 15 min of flight were used to go through a series of test cards for system checkout and verification of the PiccoloPlus autopilot system. After successfully passing all of the test cards, interaction of the VC and Ares-2 was tested by manually commanding waypoint target changes to cause the UA to switch from flight plan 1 to flight plan 2. Finally, two separate experiments were run to test the two primary triggers laid out in the test plan: temperature and communication link.

During the experiment, the MNR on the UA continually downlinked packets to the RMS, which included the Thalassa sensor measurements, GPS position from the PiccoloPlus, and waypoint commands generated by the AFC Naiad. Thalassa sensor measurements were downlinked over the 802.11 DSR network to the RMS at 2 Hz and logged by the VirtualCockpit while the PiccoloPlus-specific data packets were downlinked at 1 Hz.

1. Sensor-Reactive and Communication-Reactive Control

The primary results of the flight experiment are shown in Fig. 12. In the top of the figure is the waypoint number the UA was tracking at that moment. The middle plot shows the temperature measured on board the UA (including offset), and the bottom plot shows the ping times recorded by the ground station. The horizontal time axis is referenced in mission time, which started before takeoff when the UA is powered and first connects to the RMS. These three plots together provide the results of executing the experimental test plan described above.

In the top plot, it can be seen that at the start experiment commands (there are two shown in the time scale covered), the UA transitions from flight plan 1 (Waypoints 2–7) to flight plan 2 (waypoints 10–15). Upon detecting a sensor trigger, as defined in the test plan, the UA transitions from flight plan 2 back to flight plan 1.

Figure 12 shows that the first trigger occurred when the temperature probe reached 40 °F. This is seen in the temperature plot (middle) with the first start experiment command sent at roughly 58 min. After the start command is given, the Thalassa node introduces an artificial offset to represent dropping temperature. Just before the 60th minute the temperature reaches 40 °F, causing the sensor trigger. The top plot shows that the waypoint being tracked jumps from waypoint 14 to waypoint 2.

The second sensor experiment was started after the 64th minute, and represents the ability of the UA to respond to changes in network performance. For this experiment, connectivity between the UA and the RMS was chosen to be the primary trigger and was measured by sending ping commands between the two. The loss of ping commands is seen in the bottom plot, with the trigger occurring 40 s after the last ping was received.

2. UAS-RMS Network Communication

For the experiments, user datagram protocol (UDP) socket connections were used to link the RMS to the UA. Figure 13 shows the round-trip ping times from the RMS to the UA back to the RMS during system checkout on

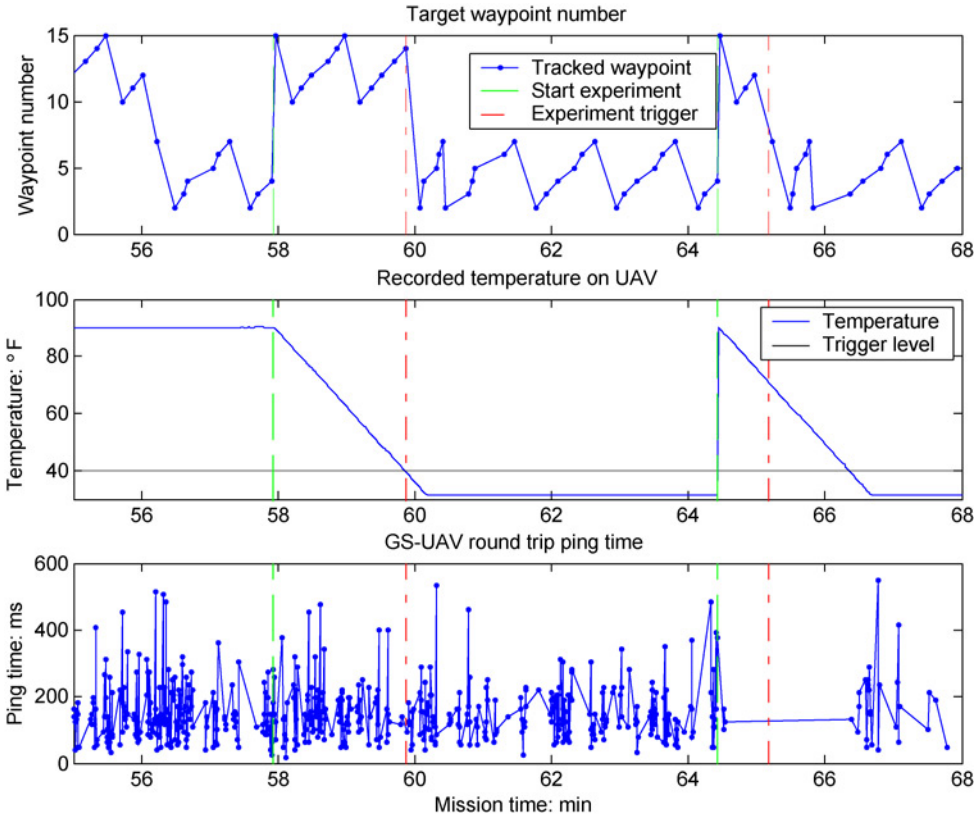


Fig. 12 Experiment results.

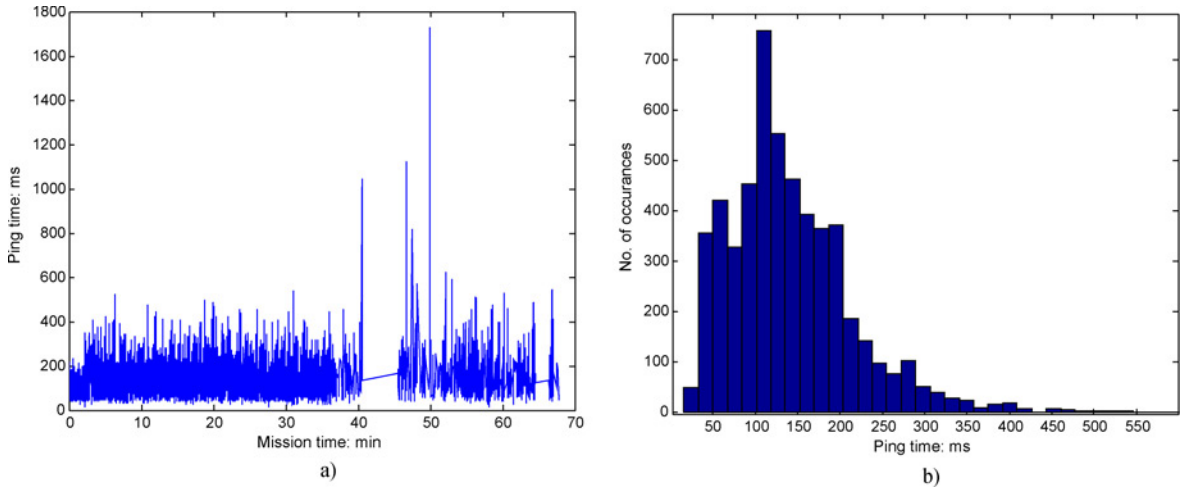


Fig. 13 Round-trip ping times: a) ping times vs mission time, b) histogram.

the ground and through the flight. Figure 13a shows the ping time for the duration of the mission and Fig. 13b is a histogram that bins the ping time according to round-trip duration. The ping times are typically 120 ms, which includes the processing time on each end and transmission through one or more intermediate mesh network links.

From previous flight experience, it was believed that the ping times would be dependent upon the attitude of the aircraft relative to the ground station. Owing to the close operation of the aircraft to the ground station (typically within 1 km), however, Fig. 14 shows that there is no noticeable dependency of the ping time on location (which relates to a turning attitude within the flight plan). The dense cluster of ping times at the center of the plots represents the data collected while the UA was on the ground during preflight checkout. It is expected that as the operational range is increased, the ping times will be affected by the attitude of the aircraft. In addition, when a multihop link is used, the ping time will again be dependent upon location of the aircraft since location will be the primary factor in determining network routes.

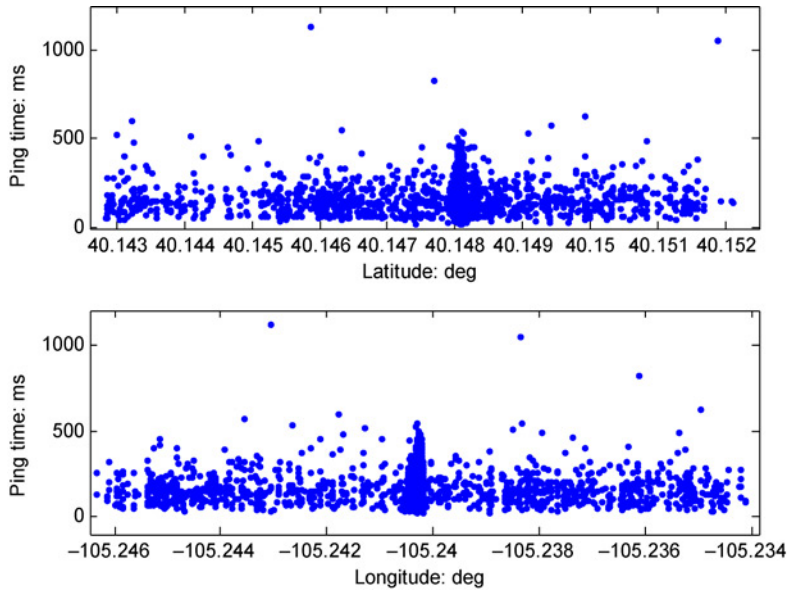


Fig. 14 Round-trip ping times plotted against GPS locations.

3. Atmospheric Sensor Measurements

Figure 15 shows plots of the relative humidity, pressure, and temperature over time that was logged by the VirtualCockpit for the entire experiment. Between the 0th and 36th minute of mission time, the change in the recorded sensor measurements is primarily a result of the warming atmosphere (from the 11:00 am to 12:00 pm hours) and transitions of the UA into and out of shade during ground setup and checkout.

The dramatic change that occurs at the 36th minute represents the takeoff phase of the experiment and shows the UA cooling to atmospheric temperature after sitting in the sun. The drop in atmospheric pressure, shown in the middle plot, is attributable to the location of the pressure sensor within the UA and is related to the airspeed of the UA.

Figure 16 shows that the humidity sensor was able to detect a change in relative humidity along the UA flight plan encircling the runway. The positional dependency of humidity over a 4-min period, starting at 62-min mission time, is shown. The left plot shows the GPS location of the UA colored by the measured relative humidity over an aerial image of Table Mountain. Black represents a relative humidity below 14% while red is for 16.5% and above. The colors change at 0.5% increments within these two bounds. The cause of the humidity change over the orbit is due to the local ground cover and terrain beneath the UA, while it is flying only 300 ft above the ground. The blue and green markers occur on the map over the Table Mountain mesa while the red and pink colors are off the mesa. Owing to the age of the aerial image, the foliage that is present on the edge of the mesa, giving the higher humidity

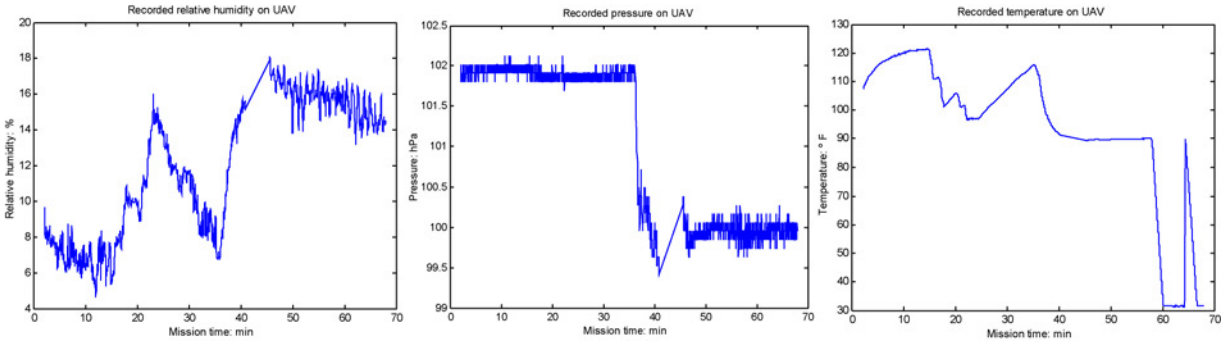


Fig. 15 Thalassa sensor measurements onboard Ares UA (takeoff is at minute 36).

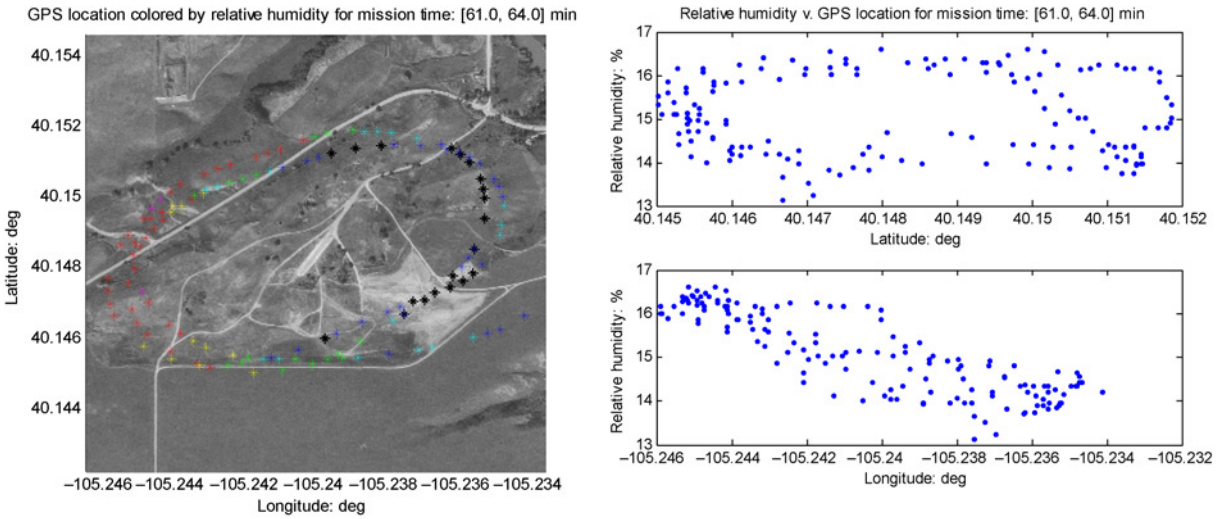


Fig. 16 Relative humidity measurements collected over a 3-min period.

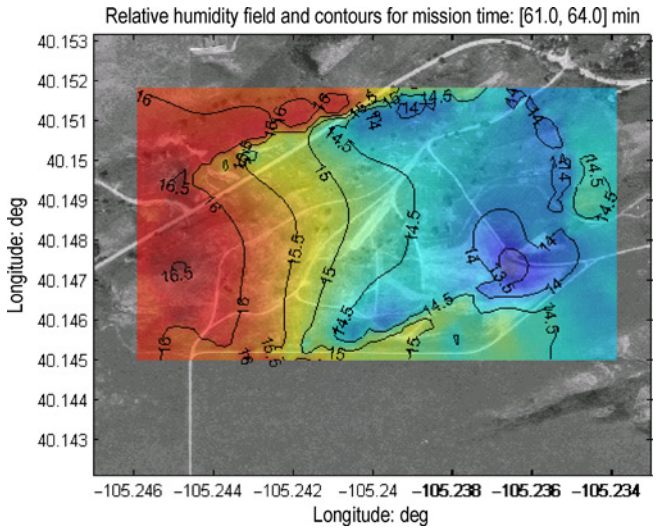


Fig. 17 Contour plot of relative humidity overlaid on aerial image of Table Mountain.

values, is not shown. In addition, a stream (that runs adjacent to the main diagonal road in the image) that contributed to the higher humidity, is not readily visible.

A contour plot was generated from these data by using a 2-D interpolation technique known as Kriging [27] and is shown in Fig. 17. This plot highlights the fact that in situ, volumetric sensing using the NetUASC3 system can provide valuable atmospheric science information.

B. Radio Propagation Environment

Additional flight testing was conducted experimentally to validate radio propagation models for the wireless network transmissions generated by the NetUASC3 system. Consider a set of M radio emitters located at positions $\boldsymbol{p}_j = (x_j, y_j)$ in the environment. The logarithm of the power $P_j(\boldsymbol{p}_r)$ received at position $\boldsymbol{p}_r = (x_r, y_r)$ from the jth emitter using the standard empirical radio propagation model is

$$\log P_j\left(\boldsymbol{p}_r\right) = \log\left(P_0 \cdot \left(\frac{d_0}{d}\right)^e\right) + v = k_0 - e \cdot \log d + v \tag{1}$$

where P_0 and d_0 are reference power and distance, $d = \sqrt{(x_r - x_j)^2 + (y_r - y_j)^2}$ is the distance between the emitter and receiver, $2 < e < 6$ is the propagation decay exponent, assuming e is constant in the environment, $k_0 = \log P_0 \cdot d_0^e$ lumps the constant parameters together, and v is a randomly distributed noise factor representing the radio variability. The variable $P_{i,j}$ denotes the power of the jth emitter received at the location of the ith aircraft.

Each UA in the NetUASC3 system is equipped with a receiver that can detect and measure the power of radio transmission signals. In some instances the receivers can only measure the total power received and the measurement z_i taken by the ith aircraft is

$$z_i = \sum_{j=1}^{M} P_{i,j} \tag{2}$$

In other cases the receivers can distinguish between transmissions from different sources in which case the measurement taken by the ith aircraft is the vector

$$\boldsymbol{z}_i = (P_{i,1}, \cdots P_{i,M})^T \tag{3}$$

The Atheros chipsets along with the Linux MadWifi drivers used in the NetUASC3 MNR nodes provide a measure of the received signal power for each individual link in the ad hoc network as the received signal strength indicator

(RSSI) on a packet-by-packet basis. As defined by the IEEE 802.11 standard, the RSSI is a single byte value used to represent a relative measure of RF energy as defined by the manufacturer of the specific chipset (note, not all values from 0–255 are required to be used). However, the Atheros chipsets have a defined conversion from RSSI to decibels (and therefore watts), enabling measurement and comparison of the received signal strength on an absolute scale. A limitation of the Atheros chipsets in measuring the RSSI value is the fact that the chipset utilizes only the range of 0 to 60 to report the RSSI, limiting the resolution of the measurement.

Experimental verification and fitting of the empirical propagation model was done in two separate experiments to understand and compare the RSSI measurement of a link between two quasi-static (slowly moving) ground nodes and between a static ground node and a node in an Ares UA (fast moving). In both experiments the RSSI measurement was collected by each of the two nodes for that specific link while still part of a three-node ad hoc network, where the third node is the RMS running the VC. In addition to measuring and time stamping the RSSI, latitude, longitude, and altitude data of the two nodes were collected using the GPS receivers built into the MNRs so that the distance between the two nodes can be calculated.

Figure 18 shows the results of RSSI measurements versus separation distance for two ground based MNR nodes, used in the NetUASC3 project, placed on top of 8-ft ladders. In this experiment, the empirical model was found to have parameters $k_0 = 3822.0$ and $e = 3.582$, using a least squares best-fit line of the data to the linear form of Eq. (1). The radio variability has a standard deviation $\sigma = 1.96$ dBm. Figure 18a shows that for the case of two static nodes, the power received at node j is equal to that measured at node i, i.e., $P_{i,j} = P_{j,i}$. Figure 18b indicates that while the empirical model fits the measured RSSI data in the middle range of the system (from around 500 m to 1500 m), the model begins to deviate from the measured data when nearby or far away. When the two nodes are close to each other, multipath from the ground has a significant impact on the variability of the RSSI measurement, which is seen on the left of Fig. 18a. As the separation distance increases, then the sensitivity of the 802.11 interface card comes into play and limits the minimum measured RSSI value. In the case of the Atheros chipsets used in these experiments, the receive sensitivity for the receiver is −96 dBm, so only signal strengths at approximately −96 dBm or greater are measured skewing the average measured signal strength to higher values.

Figure 19 shows the results of the second experiment using a static ground MNR on an 8-ft ladder and an MNR mounted in an Ares UA flying an oval pattern approximately 200 m above the ground. Figure 19a shows the RSSI of the UA–MNR link as measured by the UA and the MNR vs time from takeoff (far left) to landing (far right) over a 20-min span while Fig. 19b shows the RSSI as a function of distance. For this experiment, the empirical

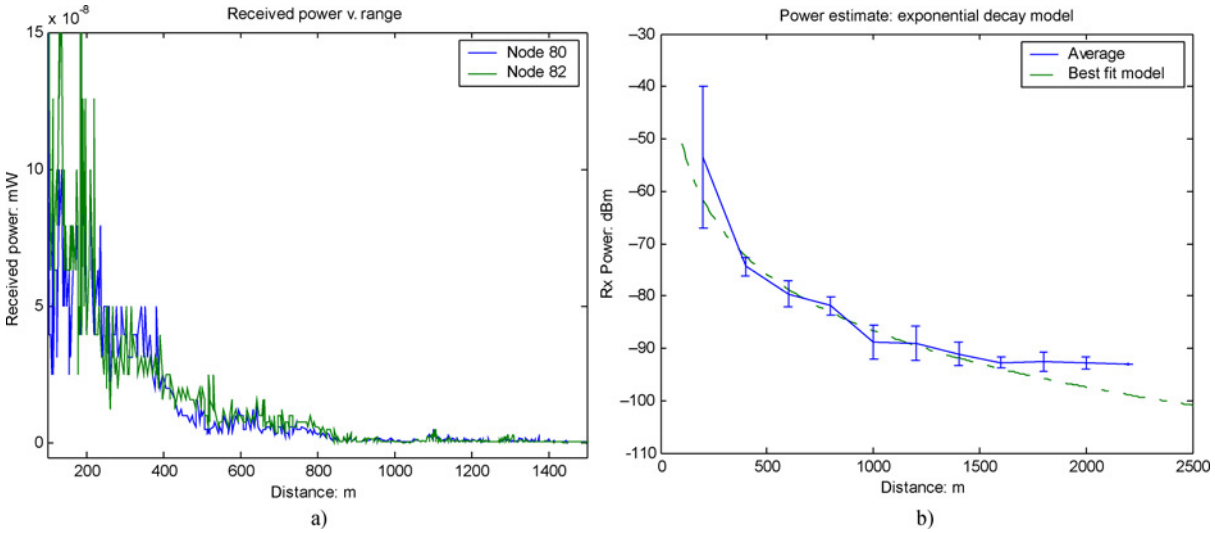


Fig. 18 Measurements of received radio power vs distance for ground-based NetUASC3 meshed radio nodes: a) raw data, b) exponential model fit.

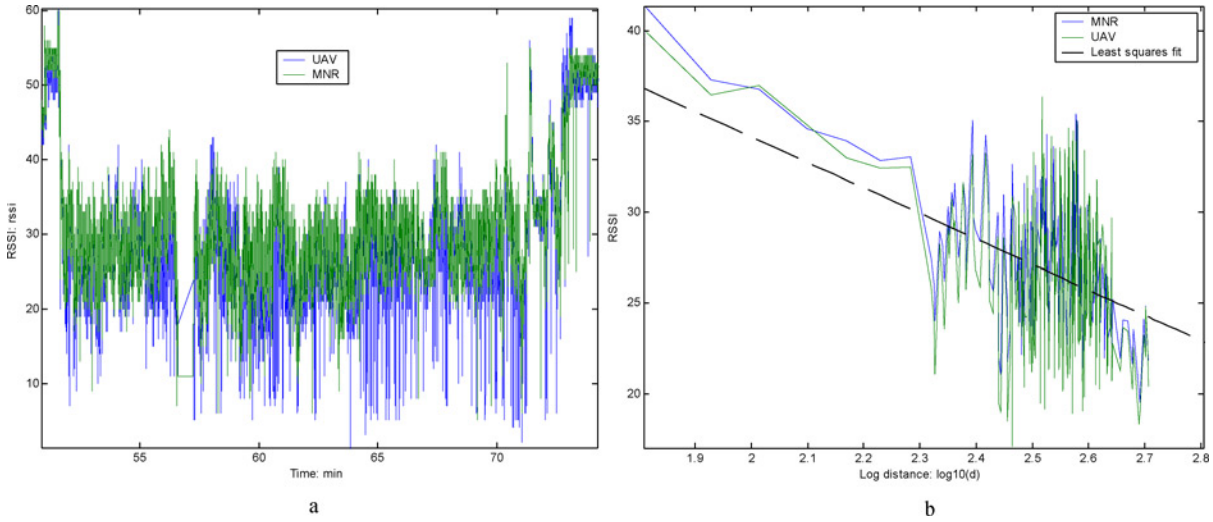


Fig. 19 Measurements of RSSI values between a UA-MNR link: a) raw data measured during the flight, b) exponential model fit.

model was found to have parameters $k_0 = 1.6528 \times 10^5$ and $e = 2.526$, while the radio variability has a standard deviation $\sigma = 3.46$ dBm.

The results of the second experiment are consistent with the conditions of the experiment in that the overall variability of the RSSI measurements has increased with the increased mobility and dynamics of the UA and the exponential propagation loss is decreased owing to elevation of the airborne UA. Interestingly, Fig. 19a shows that, while on average the UA and MNR measured the same RSSI, the RSSI measurement on the UA has a much wider variance than the ground-based MNR. This is expected since the UA operates in a higher RF noise environment than the isolated MNR on the ground. Specifically, the altitude of the UA increases interference from other 802.11 networks in the area surrounding the test range. Also, the orbiting nature of the UA during the experiment can be seen in the RSSI data as the major oscillations, particularly in the 60 min to 70 min experiment time range. Figure 20 shows a contour plot of the RSSI field as measured by the UA using the same 2-D Kriging interpolation method employed in Sec. III. A.2. While Fig. 20 shows the dropoff of RSSI with distance, it also shows that, owing to environmental effects, the field is not circular.

C. Webcam Imagery

To demonstrate another type of sensor and the ability to access sensor subsystems across platforms, a video camera was integrated into the UAS. Two different flights where made with an IP-based web camera (Panasonic BB-HCM311 Pan/Tilt IP camera) mounted on the belly of Ares-2 (the full video clips are available at http://augnet.colorado.edu). The webcam has a built-in webserver and can be viewed over an IP-based network by multiple clients. The video can be displayed, or analyzed, by making a connection to the aircraft on port 80 (see Sec. II. B) from any client or subsystem on the AUGNet. The camera has three different resolution settings of 640×480, 320×240 (used in flight test), and 160×120 with a maximum frame rate of 12 fps (frames per second) at 640×480 and up to 30 fps at the lower resolutions. The camera can pan $\pm 60°$ and tilt from $-45°$ to $+20°$, and consumes 6W during a pan/tilt scan. During the flight tests the frame rate varied and appeared to correlate with UA position and attitude. This variation in frame rate can be seen in the full video clips.

Figure 21 shows three different snapshots from the video downlinked over the AUGNet system to the VC. Figure 21a is a snapshot of an overflight of the ground control station (GCS) and a storage trailer used at the Table Mountain test site. Figure 21b and Figure 21c were taken on a second flight. The middle snapshot is of the Heiligenschein, which is the bright spot around the shadow of the aircraft. The final image was taken a minute later in the flight and shows a road construction crew on one of the side roads on the edge of the test site.

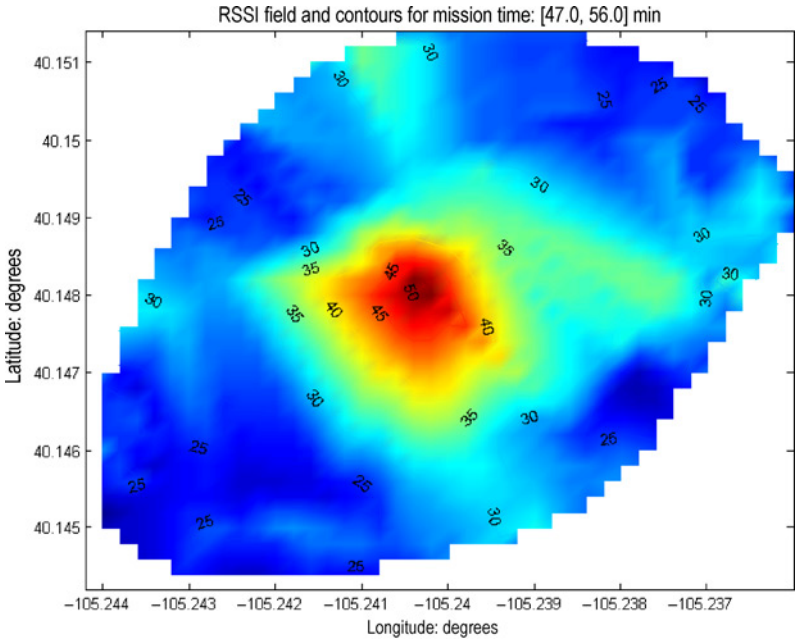


Fig. 20 Contour plot of RSSI field between UA and ground-based MNR.

Fig. 21 Images from a webcam on board Ares-2 from two different flights, highlighting items of interest. Timestamps of images are shown in upper left boxes: a) Over flight of ground control station (GCS) with two vehicles and storage trailer, b) image of Ares-2 shadow, c) over flight of road construction crew.

For the implementation presented here, the video was downlinked over the network as a series of JPEG images, not in a streamed format such as MPEG. This was beneficial for several reasons including image quality and ease of use by clients in the network. While using an MPEG stream may improve the frame rate in a relatively static network over a few hops, in a dynamic environment with rapidly changing routes the MPEG stream may become stagnant and continually try to resynchronize the stream or lower the video quality. By using a series of JPEG images instead, when a new link is established only the new images which are relatively small need to be sent on the link and there is no resynchronizing of the data stream. In addition, other clients on the network such as the display component of the GCS can process single JPEG images more easily than an MPEG stream.

Since video imagery is stamped with system time, video quality and frame rate can be correlated with other telemetry data. Figure 22 shows video frame rate correlated with aircraft roll angle. The frame rate is calculated by averaging over a 1.0 s window every 1.0 s. The roll angle measurement is provided by the PiccoloPlus autopilot over a different telemetry stream than the video; however they are both time-stamped from the same system clock. Data

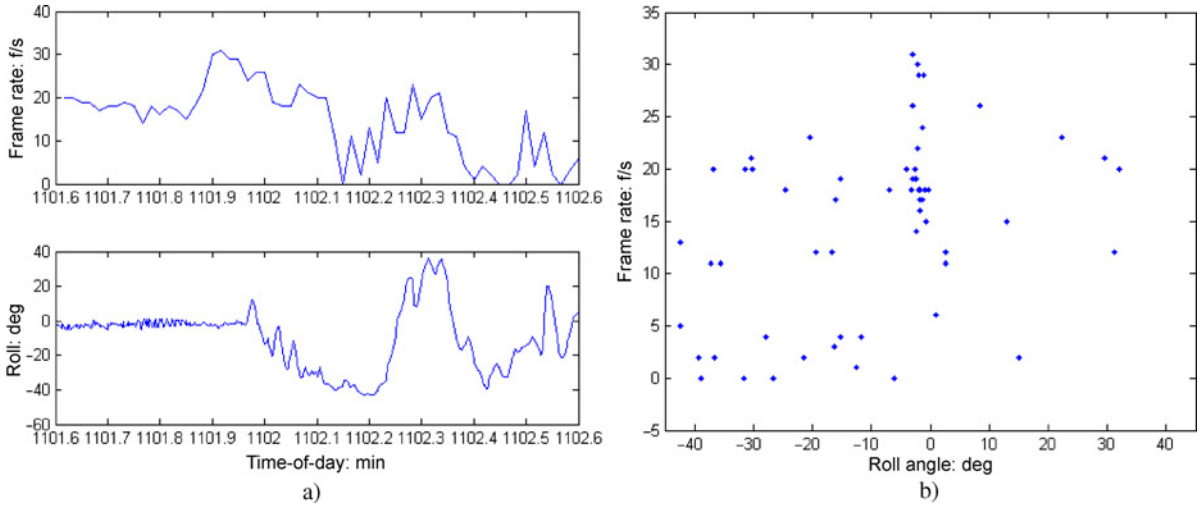


Fig. 22 Frame rate data correlated with aircraft roll angle: a) frame rate vs time and roll angle vs time, b) frame rate vs roll angle.

in Fig. 22a show a nominal frame rate of approximately 20 fps while the aircraft sits on the ground. This figure is less than the 30 fps maximum because the network quality is reduced when so close to the ground. After takeoff (approximately 1109.5 min) the aircraft banks away from the ground station and the frame rate drops significantly. The UA levels out and the frame rate improves. Figure 22b shows the correlation between aircraft roll angle and frame rate. Although the data show variability, the average frame rate decreases for larger roll angles. Finally, Fig. 23 shows the frame rate along the flight path. Color markers are used to indicate the frame rate at a given UA location on the map. The roll angle dependence can be seen during the various turns (the transmission antenna is located under the belly of the UA so the clockwise turns point it away from the ground station). The drop in frame rate at the end of the path in Fig. 23 (lower left corner of the image) is because the UA is descending.

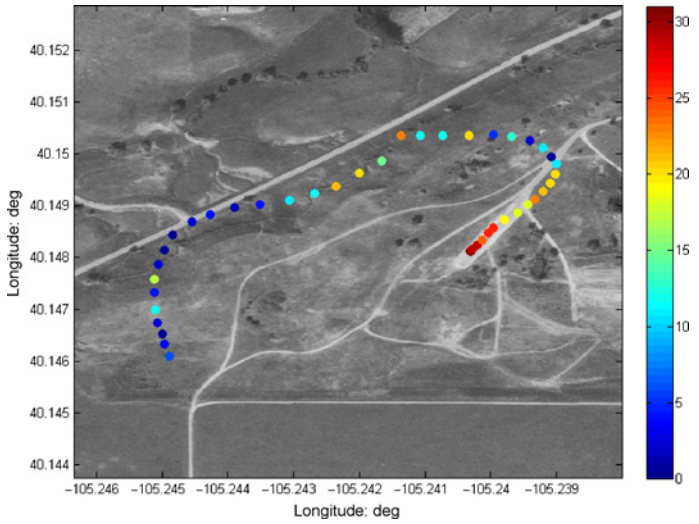


Fig. 23 Video frame rate determines marker color for UA position overlay.

IV. Conclusion

The present paper described the NetUASC3 architecture motivated by the fact that fully autonomous, cooperative, multivehicle operation requires the development and integration of various levels of intra- and intervehicle communication, sensing, control, and autonomy. The NetUASC3 architecture combines the AUGNet mobile ad hoc network, the Ares unmanned aircraft, and a new onboard flight management architecture to enable command and control of multiple unmanned aircraft from multiple dispersed operators over a meshed ad hoc network. Control systems that allow for autonomous networked operation are integrated into existing UAS platforms. Individual UA subsystems, including automatic flight control and payload sensors, are connected through two onboard subnets to supervisory control and external IP-based mesh networking. Software algorithms that tie sensor measurements, network intelligence (i.e. RF signal level, data throughput, and so on) and mission-level tasking information into automatic flight controls refine UAS control systems and software.

Multi-UA flight operations are further supported by the NetUASC3 architecture through tight integration of the onboard flight management system with the external AUGNet system. Network address translation allows for IP-based communication between individual subsystems within the UA system. In other words, the NetUASC3 enables both meshed command and control of multiple UA from multiple operators and multihop intervehicle communication, e.g., between the supervisor control computer of one UA, the sensor payload of another, and the flight control subsystem of a third. The "bottom-up" approach to tactical-level, multivehicle UAS development presented here contrasts the "top-down" approach often taken with collaborative multi-agent system design (i.e. design the cooperation algorithms first and add the networking second) and provides a hardware base for a multi-UA network for missionizing group-UAS applications.

Results from flight demonstration of the NetUASC3 architecture validated its performance and highlighted the interplay between the various subsystems. At all times, communication, command, and control of the UA during flight testing were conducted over the meshed ad hoc network described here (although other communication systems were functioning as safety backups). During these flights a single unmanned aircraft was able to respond successfully to (simulated) adverse sensor conditions and communication loss, demonstrating the autonomous interplay between various subsystems including supervisory control, flight control, communication, and external (payload) sensing. Additional results illustrated meshed network performance, atmospheric data collection, validation of radio-propagation models, and delivery of streaming video over a multihop airborne–ground network.

Limitations of the current architecture stem mainly from the highly dynamic nature of UAS operations. In particular, standard meshed networking protocols assume end-to-end connectivity in order to establish a communication link. New fault-tolerant networking concepts will expand the capability of the NetUASC3 to enable successful transmission of data from source to destination regardless of network connectivity at any one time. Service discovery will also be needed in order to accommodate the dynamic composition of the unmanned aircraft system. As vehicles join and leave the team, member capabilities must be known before intervehicle communication and cooperation is possible. Likewise, new mechanisms for remote management by multiple dispersed operators are required to take full advantage of the NetUASC3 system.

Immediate future plans will focus on validation of multiflight operations. Issues that will be addressed include service discovery within the vehicle team, fault-tolerant networking, remote management of the UAS, and flight demonstrations. Hardware-in-the-loop testing will first demonstrate operation of the system before full deployment. Additionally, new interfaces are under development to allow remote management of the UAS from dispersed operators over the Internet. These interfaces are based on the extensible markup language (XML) in order to generalize them to various unmanned systems, missions, payloads, etc. Finally, multi-UA flight operations will be conducted in the near future once a Certificate of Authorization (COA) from the Federal Aviation Administration is obtained. Future applications of the NetUASC3 include persistent surveillance, target tracking, cooperative radio source localization, and controlled mobility for airborne networking.

References

[1] Weatherington, D., "Unmanned Aerial Vehicles Roadmap: 2002–2027," Office of the Secretary of Defense, Dec. 2002.

[2] Argrow, B., Lawrence, D., and Rasmussen, E., "UAV Systems for Sensor Dispersal, Telemetry, and Visualization in Hazardous Environments," *43rd Aerospace Sciences Meeting and Exhibit*, AIAA, Reno, NV, 2005, pp. 15097–15107.

[3] Holland, G. J., Webster, P. J., and Curry, J. A., "The Aerosonde Robotic Aircraft: A New Paradigm for Environmental Observations," *Bulletin of the American Meteorological Society*, Vol. 82, No. 5, 2001, pp. 889–901.
doi: 10.1175/1520-0477(2001)082<0889:TARAAN>2.3.CO;2

[4] Curry, J. A., Maslanik, J., and Holland, G., "Applications of Aerosondes in the Arctic," *Bulletin of the American Meteorological Society*, Vol. 85, No. 12, 2004, pp. 1855–1861.
doi: 10.1175/BAMS-85-12-1855

[5] Schulze, K., and Buescher, J., "A Scalable, Economic Autonomous Flight Control and Guidance Package for UAVs," *2nd AIAA "Unmanned Unlimited" Sytems, Technologies, and Operations Conference and Workshop*, AIAA, San Diego, CA, 2003, pp. 1–10.

[6] Frew, E., Spry, S., and McGee, T., "Flight Demonstrations of Self-Directed Collaborative Navigation of Small Unmanned Aircraft," *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop, and Exhibit*, AIAA Chicago, IL, 2004, pp. 1108–1121.

[7] Chaimowicz, L., Grocholsky, B., and Keller, J.F., "Experiments in multirobot air–ground coordination," *2004 IEEE International Conference on Robotics and Automation*, 26 April–1 May 2004, Vol. 4, New Orleans, LA, pp. 4053–4058.

[8] How, J., King, E., and Kuwata, Y., "Flight demonstrations of cooperative control for UAV teams," *Collection of Technical Papers – AIAA 3rd "Unmanned-Unlimited" Technical Conference, Workshop, and Exhibit*, Vol. 1, AIAA, Chicago, IL, 2004, pp. 505–513.

[9] Nelson, D. R., McLain, T. W., and Christiansen, R. S., "Initial Experiments in Cooperative Control of Unmanned Air Vehicles," *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop, and Exhibit*, AIAA, Chicago, IL, 2004, pp. 666–674.

[10] Ryan, A., Zennaro, M., and Howell, A., "An Overview of Emerging Results in Cooperative UAV Control," *Proceedings of the 43rd IEEE Conference on Decision and Control*, IEEE, Paradise Island, Bahamas, 2004, pp. 602–607.

[11] Fax, J. A., and Murray, R. M., "Information Flow and Cooperative Control of Vehicle Formations," *IEEE Transactions on Automatic Control*, Vol. 49, No. 9, 2004, pp. 1465–1476.
doi: 10.1109/TAC.2004.834433

[12] Leonard, N. E., and Fiorelli, E., "Virtual Leaders, Artificial Potentials and Coordinated Control of Groups," *40th IEEE Conference on Decision and Control*, Orlando, FL, 2001, pp. 2968–2973.

[13] Dunbar, W. B., and Murray, R. M., "Model Predictive Control of Coordinated Multivehicle Formations," *Proceedings of IEEE Conference on Decision and Control, 10–13 Dec. 2002*, Vol. 4, Las Vegas, NV, 2002, pp. 4631–4636.

[14] Reynolds, C. W., "Flocks, Herds and Schools: A Distributed Behavioural Model," *Computer Animation: CG87. Proceedings of the Conference held at Computer Graphics 87, Oct. 1987*, Online Publications, London, UK, 1987, pp. 71–87.

[15] Jadbabaie, A., Lin, J., and Morse, A. S., "Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules," *IEEE Transactions on Automatic Control*, Vol. 48, No. 6, 2003, pp. 988–1001.
doi: 10.1109/TAC.2003.812781

[16] Olfati-Saber, R., "Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory," *IEEE Transactions on Automatic Control*, Vol. 51, No. 3, 2006, pp. 401–420.
doi: 10.1109/TAC.2005.864190

[17] Bellingham, J. S., Tillerson, M., and Alighanbari, M., "Cooperative Path Planning for Multiple UAVs in Dynamic and Uncertain Environments." *41st IEEE Conference on Decision and Control, 10–13 Dec. 2002*, Vol. 3, Institute of Electrical and Electronics Engineers Inc., Las Vegas, NV, 2002, pp. 2816–2822.

[18] Brown, T. X., Argrow, B., and Dixon, C., "Ad Hoc UAV Ground Network (AUGNet)," *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, AIAA, Chicago, IL, 2004, pp. 29–39.

[19] Brown, T. X., Doshi, S., and Jadhav, S., "Test Bed for a Wireless Network on Small UAVs," *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop, and Exhibit*, AIAA, Chicago, IL, 2004, pp. 432–439.

[20] Dixon, C., Frew, E. W., and Argrow, B., "Radio Leashing Unmanned Aircraft," *AIAA Infotech@Aerospace*, AIAA, Arlington, VA, 2005, pp. 1093–1102.

[21] Frew, E., Dixon, C., and Argrow, B., "Radio Source Localization by a Cooperating UAV Team," *AIAA Infotech@Aerospace*, AIAA, Arlington, VA, 2005, pp. 10–20.

[22] Frew, E. W., and Lawrence, D., "Cooperative Stand-off Tracking of Moving Targets by a Team of Autonomous Aircraft," *AIAA Guidance, Navigation, and Control Conference*, AIAA, San Francisco, CA, 2005, pp. 4885–4895.

[23] Brown, T. X., Doshi, S., and Jadhav, S., "A Full-Scale Wireless Ad Hoc Network Test Bed," *Proceedings of International Symposium on Advanced Radio Technologies*, Boulder, CO, 1–3 March, 2005, pp. 1–10.

[24] Jadhav, S., Brown, T. X., and Doshi, S., "Lessons Learned Constructing a Wireless Ad Hoc Network Test Bed," *Proceedings of the Wireless Network Measurement Workshop*, Trentino, Italy, 2005, pp. 1–6.

[25] Elston, J., Argrow, B., and Frew, E., "A Distributed Avionics Package for Small UAVs," *AIAA Infotech@Aerospace*, AIAA, Arlington, VA, 2005, pp. 733–742.
[26] Cloud Cap Technology, Inc., "*Piccolo Plus Autopilot*," [online database], http://www.cloudcaptech.com/piccolo_plus.shtm [accessed 9 April 2008].
[27] Oliver, M. A., and Webster, R., "Kriging: A Method of Interpolation for Geographical Information Systems," *International Journal of Geographical Information Systems*, Vol. 4, No. 3, 1990, pp. 313–332.

Kalmanje Krishnakumar
Associate Editor