ICCSwarm: A Framework for Integrated Communication and Control in UAV Swarms

Jonathan Diller, Peter Hall, Corey Schanker, Kristen Ung, Philip Belous, Peter Russell and Qi Han
Colorado School of Mines, Golden, Colorado, USA
{jdiller, peterhall, ckschanker, kung, pbelous, prussell, qhan}@mines.edu

ABSTRACT

Swarms of Unmanned Aerial Vehicles (UAVs) have many applications including search and rescue, disaster response, surveillance, infrastructure inspection, among many others. A key aspect of UAV swarms is keeping the UAVs connected through wireless communication and any deployment of UAV swarms must consider communication constraints during motion planning. In this paper we introduce *ICCSwarm*, a framework for integrating communication and control in UAV swarms. *ICCSwarm* consists of two phases, a planning phase where combined communication and motion planning are validated in simulation, and a deployment phase, where a UAV architecture designed around integrated communication and control executes missions from the planning phase on physical UAVs. We implemented *ICCSwarm* on a physical UAV testbed and evaluated its effectiveness through a unique case study in partnership with NASA's JPL. We deployed the UAVs to trial small satellite orbits for data collection on asteroids and the results validate our design and highlight *ICCSwarm*'s capabilities.

CCS CONCEPTS

• **Computer systems organization** → **Robotics**; • **Networks** → *Mobile ad hoc networks.*

KEYWORDS

UAV swarm, UAV networks, autonomous UAVs, UAV data collection

ACM Reference Format:
Jonathan Diller, Peter Hall, Corey Schanker, Kristen Ung, Philip Belous, Peter Russell and Qi Han. 2022. *ICCSwarm*: A Framework for Integrated Communication and Control in UAV Swarms. In *Eighth Workshop on Micro Aerial Vehicle Networks, Systems, and Applications (DroNet '22), July 1, 2022, Portland, OR, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3539493.3539579

1 INTRODUCTION

Unmanned Aerial Vehicle (UAVs) have been proposed for many of today's real world problems. UAVs can traverse over rough terrain and water, they can operate during infrastructure failure, they are commercially available and affordable, and they can easily go to places that are challenging for humans to reach. Applications for UAVs include search and rescue, disaster response, surveillance, infrastructure inspection, among many others. Using swarms of UAVs has the potential to expedite these tasks by having multiple UAVs work as a team to perform tasks in parallel.

Wireless communication is one of the major challenges in implementing and deploying UAV swarms. Moving data through a wireless network of UAVs requires the UAVs to coordinate connectivity while moving through the environment. Updating task allocation during a mission execution may require the UAVs to maintain constant communication with a base station on the ground. This level of coordinated control across UAVs requires a mobile, dynamic, ad-hoc network that can self-configure as the UAVs move around and mission planning must consider the requirements of such a network.

To address this need for integrated motion planning and networking we propose *ICCSwarm*, a framework for integrating communication and control in UAV swarms. *ICCSwarm* consists of two phases, a planning phase where we combine communication and path planning offline, and a deployment phase where missions from the planning phase are performed on physical UAVs. We implemented *ICCSwarm* on a physical UAV testbed and validated our design with a novel UAV application where we mimic swarms of small spacecraft for scientific data collection around an asteroid.

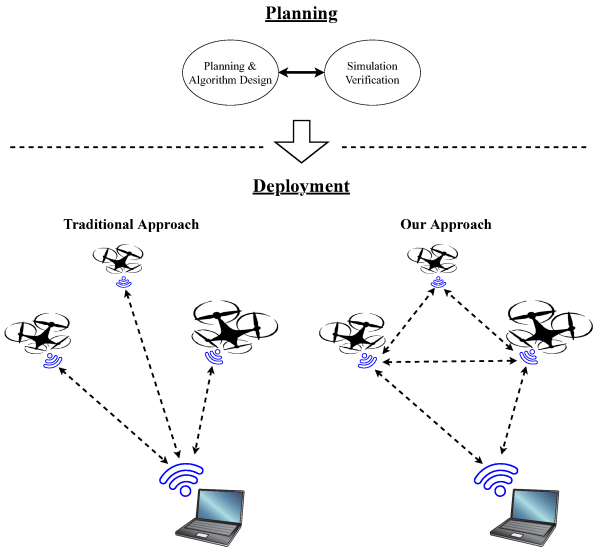


Figure 1: Two-phase approach to planning and deploying UAV swarms showing our design compared to the traditional approach.

DroNet '22, July 1, 2022, Portland, OR, USA

ACM ISBN 978-1-4503-9405-5/22/07…$15.00
https://doi.org/10.1145/3539493.3539579

2 RELATED WORK

In recent literature there have been many UAV testbed designs. For instance, the authors in [8] introduce a framework called Autonomous, Sensing, and Tetherless netwoRked drOnes (ASTRO), which is designed for networked, tetherless, on-line decision making teams of UAVs. ASTRO is similar to *ICCSwarm*, except that it is data-centered decision making where *ICCSwarm* makes networking and communication a more central aspect of planning. ASTRO was later extended for sensing and monitoring air pollution in [2]. Many UAV swarm testbeds have been built to focus on search and rescue applications [1, 13]. An example of an indoor testbed is presented in [3] where they use the Marvelmind indoor navigational system to help orient small UAVs within a netted arena.

Many have proposed using simulations to validate path planning designs but lack a physical test bed to validate algorithms. For instance, the authors in [14] present OpenUAV, a general purpose, cloud-based, open source UAV simulation testbed that is designed for both single and multiple UAV simulations that could be used for evaluating path planning algorithms but does not include peer-to-peer networking for the UAVs. The authors in [9] propose using a Dynamic Data-Driven Application System (DDDAS) approach to share data between simulations and improve UAV planning. In [10] they present DroneNet-Sim, a simulation framework that uses machine learning on real-world trace data to help integrate realistic networking conditions with UAV simulations.

There are also examples of integrated communication for UAV swarms but many of these works are not presented with a physical testbed to validate the design. In [5] the authors discuss different approaches to virtual networking infrastructure management with a focus on UAV applications. In [4] they propose a swarm architecture that uses both peer-to-peer communication between UAVs and cellular network connection to help distribute commands to the UAVs. In [6] the authors present a communication aware path planning algorithm that they evaluate in simulation where the UAVs rely on existing cellular infrastructure to help with connectivity.

3 SYSTEM ARCHITECTURE

Traditionally, UAVs in a testbed are wirelessly connected to a base station. This approach limits the range of the UAVs and is not dynamic. Some previous designs have proposed using cellular infrastructure to extend UAV capabilities [4, 6], but these designs are not usable in disaster scenarios where this infrastructure may be destroyed. It would be better to have a system that extends the range of the UAVs and is more dynamic, not dependent on critical infrastructure.

In contrast, we propose a more adaptive approach where the UAVs self-organize themselves into a multi-hop ad-hoc mobile network communicating with the base station. Connecting the UAVs through a mobile ad-hoc network allows the UAVs to share data amongst themselves, leading to more timely collaborative autonomy. Further, this approach extends the range of the UAVs for relaying data back and forth with the base station. Figure 1 shows this distinction.

The central base station could be a laptop using WiFi or could be further supported using cellular network infrastructure. To avoid dependency on existing infrastructure alone (such as cellular networks), we propose the flexibility of using a centralized laptop with ad-hoc WiFi in addition to existing infrastructure because it provides a more reliable setup for more applications.

3.1 Planning & Deployment Phases

We propose a two-phase approach for *ICCSwarm*, where an initial, integrated communication and control plan is formulated offline in a planning phase and then deployed and updated in real-time in a deployment phase.

In the planning phase, we design UAV path planning algorithms by taking into consideration communication constraints. Communication constraints can include restrictions such as limited communication range, enforcing constant connectivity, or considering bandwidth limitations. We then validate, refine, and evaluate the effectiveness of the planned paths through simulations that include these communication constraints. We simulate data collection and transmission across multiple moving nodes based on the results of the planning algorithm. These simulation results can then be used as feedback to further refine UAV motion planning and control. Once we have found a set of UAV paths, we generate an autopilot mission that can be executed by the autopilot software on the physical UAVs (further discussed below).

In the deployment phase, we run application-specific missions generated in the planning phase on our physical UAV swarm. This phase acts as an opportunity to validate simulation results and evaluate algorithms communication-aware UAV path planning. This phase is also designed to execute real-world applications enabled by multiple UAVs. During deployment, real-time network status data is used to update the UAV's mission in real-time to help adapt to a changing environment.

3.2 *ICCSwarm* UAV Architecture

The UAV architecture for our *ICCSwarm* framework is centered around a mission computer and flight controller. The mission computer is an on-board computer running the Linux OS with an autopilot, network routing component, and a network monitoring component. These components run together with the mission that was planned offline to perform application specific tasks. The mission computer supports wireless communication through WiFi or other application suitable wireless technology. The flight controller is a separate chip on the UAV that receives movement commands from the mission computer and interacts directly with the UAV hardware. Figure 2 shows our UAV architectural design.

The application specific mission is comprised of a UAV routing component and a data-collection and transmission component. The routing component is a sequence of movement commands that direct the UAV where to move. The data collection component acts as UAV swarm tasking, such as capturing data using on-board sensors and forwarding the data through a multi-hop network to a centralized location. These two components run separately from but parallel to one another and are joined together as a single mission. The initial mission is planned offline, but due to the communication-aware nature of our design these missions can be updated during execution. Although the initial offline plan can account for networking conditions during planning phase, the network routing

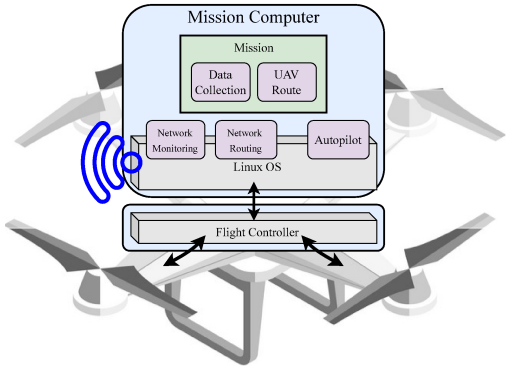


Figure 2: *ICCSwarm* UAV Architecture Design.

and network monitoring components (further discussed below) allow the UAVs to self-organize into a multi-hop network and select the best routing path for data during the deployment phase.

The autopilot is responsible for running the individual application specific missions by interfacing with the flight controller to realize UAV movement commands and running data collection. The autopilot initializes the data collection part of a mission based on how that mission is configured. The data collection component can interface directly with the Linux OS, allowing access to hardware peripherals such as on-board sensors and data transmission using the network routing component of the mission computer. The autopilot executes movement commands by interfacing with the flight controller using an API and allows for more complex movement commands to be formed from the basic commands provided in the API. The flight controller interfaces directly with the UAV hardware and handles the stability and movement of the vehicle, abstracting this level of detail away from the autopilot and mission computer.

The network routing component is used to handle multi-hop data routing and transmission through mobile ad-hoc network (MANET) routing protocols. This design allows communication and data transmission to be integrated into movement control while also being disconnected from the autopilot and low-level movement components. Disconnecting communication at this level allows us to design, develop, experiment with, and deploy different types of MANET routing protocols while also keeping communication a central part of the larger system.

The network monitoring component allows the UAV to monitor networking conditions such as received signal strength, network traffic, or radio jamming. The mission computer can use this component to make control decision updates during mission execution based on network conditions.

4 UAV SWARM IMPLEMENTATION

We implemented our *ICCSwarm* design concepts on a team of UAVs.

4.1 Hardware Implementation

For the physical testbed, we chose the widely available Raspberry Pi 3B+ (RPi) as our on-board mission computer and Emlid's Navio2 hat for the RPi as a flight controller. This system allows us to actively run test mission scripts during guided flight from a Python

Table 1: Movement Commands

Command	*Arguments*	*Description*
GainAlt	*alt*	Move the UAV to altitude *alt*
WaypointDist	*east, north, alt*	Move the UAV to waypoint described by (*east, north, alt*) arguments
WaypointTime	*east, north, alt, t*	Move the UAV towards waypoint (*east, north, alt*) for *t* seconds

autopilot script on the mission computer while easily communicating with the flight controller. The Navio2 integrates a suite of sensors, which include the global navigation satellite system receiver for GPS, dual inertial measurement units (IMU), barometer, and several ports to connect additional external sensors. Additionally, the Navio2 provides 15 pulse width modulation (PWM) and pulse position modulation (PPM) pins that allow for a relatively simple interface with the electronic speed controllers (ESC) and radio receiver, respectively. With the UAVs having a weight of approximately 1.2 kg, we paired 935KV brushless motors with 30 amp ESCs to provide a thrust to weight ratio greater than two, which gives the drones adequate maneuverability. To interface with a traditional radio transmitter we used the Flysky FS-iA6B 2.4GHz six channel receiver that can be wired into the PPM pins on the Navio2. The receiver connects to a Flysky FS-i6X radio transmitter. The transmitter has positional controls in addition to several switches which can be programmatically mapped to trigger scripts mid-flight. Figure 3 shows our physical UAV testbed.

Figure 3: Physical UAV testbed.

To power the UAV we selected an HBR 5000 mAh 14.8v Li-Poly because its capacity to weight ratio allows the UAV to have a hovering flight time of approximately 12 minutes. To finalize the testbed we used a Readytosky® S500 Quadcopter Frame with a 500mm wing span. This accommodates the mounting needed for the flight controller and the 10-inch 45-degree (1045) propellers.

4.2 Autopilot Implementation

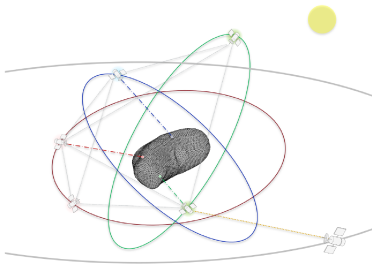
We implemented the autopilot in Python. The autopilot can respond to commands from the remote controller for the UAV, which allows

us to initialize and run missions. Currently there are several different types of movement commands implemented in the autopilot. The current command set is listed in table 1.

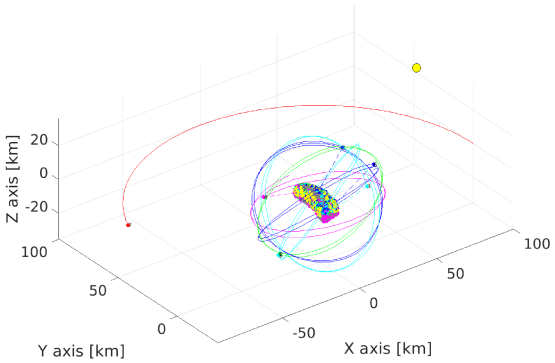
4.3 Routing Protocol Implementation

For the network routing component we implemented the Ad-hoc On-Demand Distance Vector Routing (AODV) routing protocol [7]. AODV is a routing protocol designed for mobile ad-hoc networks that need to dynamically reorganize network nodes by maintaining a routing table that is updated on-demand. When a node wants to send data through the network, it starts by sending out a route discovery message that is forwarded from node to node. If a discovery packet is received by the requested receiving node then a confirmation packet is return back to the sender. If multiple routes are discovered between the sender and receiver, the route with the shortest number of hops is used. This approach is recommended for smaller networks with moderate mobility, making it ideal for satellite orbits [12].

5 CASE STUDY: SPACE MISSION ENABLED BY SWARMS OF SMALL SPACECRAFT



(a) General SmallSat setup



(b) Output of orbit generator

Figure 4: Examples of the general problem setup for the SmallSat case study (a) and example output from orbit generator for 1.5 days of orbits (b). In (b), the larger, red orbit is the carrier satellite while the smaller, multi-colored orbits are the small satellites.

5.1 Application Background

To evaluate our system design and implementation, we used our UAV swarm as a testbed for a novel UAV application: simulating small satellite orbits around an asteroid. Many space missions involve a single spacecraft used to collect data. However, using a single spacecraft approach may limit the capabilities of the spacecraft. NASA's Near Earth Asteroid Rendezvous – Shoemaker (NEAR-Shoemaker) space probe mission is an example of this [15]. The NEAR-Shoemaker space probe was sent to study the minor planet 433 Eros, an S-class asteroid located approximately 221 million miles from Earth. Many of the sensors on NEAR-Shoemaker required different conditions to be utilized, such as spacecraft orbit altitude and light angle. These requirements forced the probe to adjust its orbit several times to properly use the different sensors and many were not usable at the same time.

To improve on the single spacecraft approach used previously, many have proposed using a swarm of small satellites to collect data in parallel [11]. In the swarm approach, each satellite has a single or a set of compatible sensors for collecting data. A larger carrier satellite acts as a central, moving base station that collects data from the smaller satellites and relays it back to Earth. Orbits for the individual small satellites can be customized to optimize the quality and quantity of data collected, as studied in [11]. We term this scenario SmallSat.

For our SmallSat case study, we evaluated the effectiveness of the more realistic and dynamic multi-hop routing against a single-hop routing approach where data is only sent directly to the carrier satellite from the smaller satellites. Figure 4 (a) shows the general problem setup.

5.2 Simulation Results

To simulate the SmallSat scenario we integrated NS-3[1], a discrete time networking simulator, with the 42 spacecraft simulator[2]. We generated orbits using the Monte-Carlo method [11] that maximizes the quality of data collected. Figure 4 (b) shows an example output of the orbit generated for 1.5 days with 6 small satellites collecting data and a single carrier satellite. After generating orbits, we fed the resulting trajectories into the 42 simulator. The 42 orbit simulator gives the (x, y, z) coordinates at discrete time intervals of the orbits for all small satellites in the swarm. These coordinates are then used by NS-3 to simulate the mobility of the networked nodes. The nodes in NS-3 generate data as if they were using sensors and attempt to send that data back to a carrier node. The NS-3 simulator provides a reliable AODV implementation.

We ran the simulation on sets of generated orbits with varying time duration, from 24 hours up to 48 hours at 3-hour increments. The satellites were set to collected data every 20 seconds. We limited on-board data storage for each small satellite at 6 hours of operation. In other words, without transmitting data to the carrier satellite, a small satellite will run out of data storage after 6 hours of operation. The carrier satellite had no data storage limit. Figure 5 shows our simulation results, graphing the total data collected over orbit duration. We found an average 240.2% increase in data collection from the single-hop approach to the multi-hop approach.

[1] https://www.nsnam.org/about/
[2] https://software.nasa.gov/software/GSC-16720-1

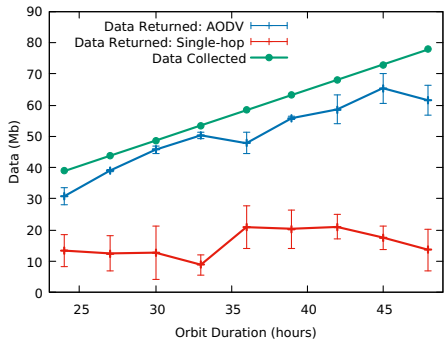


Figure 5: Simulation results: total data collected over orbit duration time.

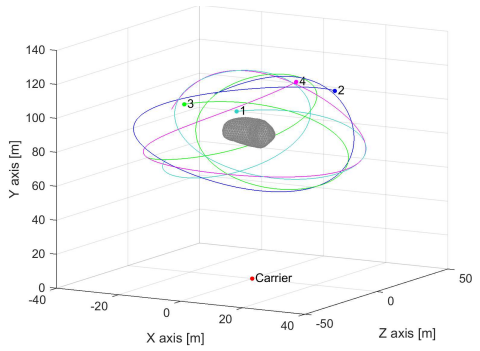


Figure 6: Transformed versions of the orbits in Figure 4 (b), scaled down to a size that UAVs can travel.

The multi-hop approach had an average data loss of 13.1% while the single-hop approach was 72.5%.

5.3 Testbed Deployment

To run the SmallSat scenario on our physical UAV swarm testbed we have the UAVs follow orbits designed for the satellites while emulating data collection and transmission back to a centralized ground station. We transformed the satellite orbits into miniature replicas that can be run on Earth using UAVs. The centralized ground station (an RPi) acts as the carrier satellite.

To create replica orbits that can be flown by UAVs, we reduce the size of the orbits and transform them from the asteroid-centered frame of reference to the carrier-centered frame of reference. By definition, the asteroid sits at the origin with the carrier satellite orbiting the asteroid on the z-plane. To transform from the asteroid-centered from of reference to the carrier-centered frame of reference, we apply a simple 3-dimensional translation and rotation around the z-axis at every time step so that the carrier sits at the origin and the asteroid sits above the carrier on the y-axis. This same translation and rotation operation is applied to each of the small satellites at each time step. To scale these transformed orbits down to a size that can be flown by UAVs on Earth, we found that a scaling factor of 7.94×10^{-4} keeps the orbits below 121 m, the max allowable altitude for UAVs in the US. We also scaled down the time of the orbits by a factor of 2.31×10^{-3} from 36 hours to 5 minutes. Figure 6 shows the results of the change of reference and scaling on the set of orbits in Figure 4 (b). We chose the size scaling factor to adhering to FAA regulation but chose a different scaling factor for time because 7.94×10^{-4} would have made the orbit duration less than 2 seconds long. The satellites in the simulator collected data once every 20 seconds while the testbed does this once every 50 milliseconds. Both attempt to send data back to the carrier immediately. If no network route to the base station was found then we buffer the data and try to establish a new route again every 2 seconds in simulation and 1 second on the testbed. To account for the differences in scaling factors and communication intervals, we only consider the percentage of data successfully returned to the carrier out of all data collected.

In real applications, the satellites will not be able to communicate with one another from opposite sides of the asteroid due to loss of Line-Of-Site (LOS). To make our field evaluation more realistic we added a LOS feature to our routing protocol. Every time one of the UAVs wants to send a data packet to another UAV, the sender evaluates its current position at that time step against the current position of the intended receiver at that time step using an orbit look-up table. If a straight line connecting the positions of the two UAVs would intersect the asteroid then the receiver UAV is removed from the AODV table of the sender and no packets are sent. This highlights the usefulness of keeping the routing protocol separate at a lower level from the autopilot and motion control.

Another limiting factor in small satellite deployment is limited transmission power to send data between the satellites. To add this limitation to our test bed, we reduced the transmit power of the RPi's WiFi. Through field testing, we found that setting the transmit power to 15 dBm using *iwconfig* replicated the simulation communication range nicely.

5.4 Field Test Results

For our field test, we generated two sets of orbits for 36 hours of data collection for three satellites and ran three UAVs on the on-Earth replicas of the orbits. We compared the on-Earth replicas against the simulation for the same two orbit sets. To keep the comparison fair, we only consider the percentage of data returned to the base station instead of comparing the actual volume of data returned. When we used the full-scale orbits described in Section 5.2 in the simulation we get 84.3% of data collected returned for orbit set 1 and 84.3% for orbit set 2 using the multi-hop approach (shown in Figure 7 as "Sim. Setting 1"). Using the Single-hop approach we only get 30.53% and 34.48%, respectively. However, in our field test results on the scale-down orbits we only received 79.1% and 42.2%, respectively, using AODV and 0.7% and 0% using single-hop. Using these results as feedback for our simulation, we adjust the transmission power setting in NS3 from 118.0 dBm (the original setting) with 116.0 dBm ("Sim. Setting 2" in Figure 7). With the simulation update, we get 73.5% and 45.9% of data returned using AODV and 12.2% and 16.0% using single-hop on the full-scale orbits. Although there is still a large gap between the simulation and field testing for the single-hop approach, we found a more realistic simulation comparison to field testing for the AODV approach.

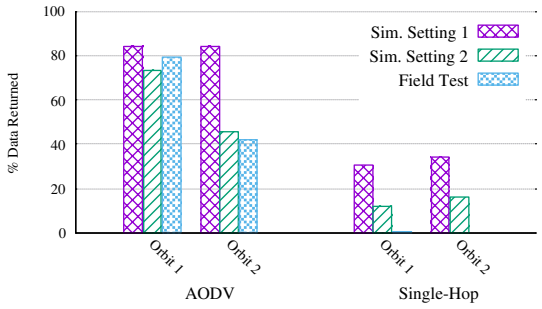


Figure 7: Simulation versus field-test results.

We suspect that the unexpected drop in performance for single-hop is because this approach uses Transmission Control Protocol (TCP) while the AODV routing protocol uses User Datagram Protocol (UDP). TCP requires receipt confirmation while UDP does not, leading the single-hop approach to be much more conservative with network conditions and under-perform while the multi-hop approach excelled at the cost of potentially losing some of the data.

6 DISCUSSION & LESSONS LEARNT

We encountered several challenges while implementing our framework. One of the main challenges was inconsistencies in UAV component performance, especially in ESCs, motors, and GPS. Some ESC and motor brands lacked sufficient quality control and were too inconsistent, leading to drone crashes. We found that EMAX motors and Hobbypower ESCs perform the most consistently. The Navio2 sensor suite sometimes experiences bad GPS positioning, causing drones to randomly fly away. Future testbed implementations could benefit from redundant GPS modules.

This project was part of an undergraduate research experience program and provided the students with an introduction to academic research. The students reported many lessons learned through the experience. Chiefly among these was learning how to isolate and identify problems and the importance of reliable and redundant components on a physical system. They also learned about designing and building physical systems, writing maintainable computer code and considering how to balance the trade-offs of different problem solutions.

There are several areas for future work. We did not implement our design for *online* coupling of control and communication in our case study and will focus on this for future work. We plan to expand the available commands for controlling the UAV in our autopilot software and find ways to make the UAV testbed more robust.

7 CONCLUSIONS

In this work we presented *ICCSwarm*, a framework for integrating communication and control on physical UAV swarms. We proposed using a two-phase approach where algorithms are designed and tested offline in simulation, then deployed on the physical UAVs. We implemented our system on a physical testbed and evaluated its design through a case study. We used our UAV swarm to compare two network routing protocols for use on small satellites for asteroid data collection. Our results show that the multi-hop approach greatly outperforms the single-hop approach, both in simulation and in field testing. Our field testing shows that the simulation is accurate for multi-hop routing but is not as accurate for a single-hop approach in this use case. We argue that this inconsistency demonstrates the need for a physical testbed to validate and provide feedback for simulation experiments. Our field test results also demonstrate the capabilities of our system for integrating communication and control. We believe that *ICCSwarm* addresses a need in literature for a UAV swarm framework that integrates communication and control that has been verified and tested on hardware.

ACKNOWLEDGMENTS

To Josh Rands and Zachary Smeton for helping implement the AODV protocol and early work on the simulations and to David Western for creating and constructing the initial UAV design.

This work was supported in part by NASA SmallSat Technology Partnership (STP) program, grant number 80NSSC18M0048.

REFERENCES

[1] Mohamed Abdelkader, Usman A Fiaz, Noureddine Toumi, Mohamed A Mabrok, and Jeff S Shamma. 2021. RISCuer: a reliable multi-UAV search and rescue testbed. In *Unmanned Aerial Systems*. Elsevier, 345–374.

[2] Ahmed Boubrima and Edward W Knightly. 2020. Robust mission planning of UAV networks for environmental sensing. In *Proceedings of the 6th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*. 1–6.

[3] Hugo Cabrita and Bruno Guerreiro. 2021. NOVA. DroneArena: design and control of a low-cost drone testbed. In *2021 International Young Engineers Forum (YEFECE)*. IEEE, 20–25.

[4] Mitch Campion, Prakash Ranganathan, and Saleh Faruque. 2018. UAV swarm communication and control architectures: a review. *Journal of Unmanned Vehicle Systems* 7, 2 (2018), 93–106.

[5] Luis F Gonzalez, Ivan Vidal, Francisco Valera, and Victor Sanchez-Aguero. 2021. A Comparative Study of Virtual Infrastructure Management Solutions for UAV Networks. In *Proceedings of the 7th Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*. 13–18.

[6] Anusha Mujumdar, Pooja Kashyap, Swarup Kumar Mohalik, and Jim Feng. 2019. Caper: A connectivity-aware path planner with regulatory compliance for UAVs. In *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 596–603.

[7] Charles E Perkins and Elizabeth M Royer. 1999. Ad-hoc on-demand distance vector routing. In *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*. IEEE, 90–100.

[8] Riccardo Petrolo, Yingyan Lin, and Edward Knightly. 2018. ASTRO: Autonomous, sensing, and tetherless networked drones. In *Proceedings of the 4th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*. 1–6.

[9] Rachael Purta, Mikolaj Dobski, Artur Jaworski, and G Madey. 2013. A testbed for investigating the UAV swarm command and control problem using DDDAS. *Procedia Computer Science* 18 (2013), 2018–2027.

[10] Chengyi Qu, Alicia Esquivel Morel, Drew Dahlquist, and Prasad Calyam. 2020. Dronenet-sim: A learning-based trace simulation framework for control networking in drone video analytics. In *Proceedings of the 6th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*. 1–6.

[11] Federico Rossi, Saptarshi Bandyopadhyay, Mark Mote, Jean-Pierre de la Croix, and Amir Rahmani. 2020. COMMUNICATION-AWARE ORBIT DESIGN FOR SMALL SPACECRAFT SWARMS AROUND SMALL BODIES. In *AIAA/AAS Astrodynamics Specialist Conference*.

[12] Elizabeth M Royer and Chai-Keong Toh. 1999. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE personal communications* 6, 2 (1999), 46–55.

[13] Jürgen Scherer, Saeed Yahyanejad, Samira Hayat, Evsen Yanmaz, Torsten Andre, Asif Khan, Vladimir Vukadinovic, Christian Bettstetter, Hermann Hellwagner, and Bernhard Rinner. 2015. An autonomous multi-UAV system for search and rescue. In *Proceedings of the First Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*. 33–38.

[14] Matt Schmittle, Anna Lukina, Lukas Vacek, Jnaneshwar Das, Christopher P Buskirk, Stephen Rees, Janos Sztipanovits, Radu Grosu, and Vijay Kumar. 2018. OpenUAV: A UAV testbed for the CPS and robotics community. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 130–139.

[15] Asif A Siddiqi. 2018. *Beyond Earth: A chronicle of deep space exploration, 1958-2016*. National Aeronautics and Space Administration, Office of Communications.