A New Air Traffic Control: Software to Safely Control One-to-Many sUAS from a Single Command and Control System

Tyris M. Audronis, and Dana L. Abramovitz, PhD MSM.

Tempest Droneworx, Inc., Houston, TX, 77079, USA

As stated in 14 CFR §107.35, a person may not operate or act as a remote pilot in command (RPIC) or visual observer in the operation of more than one sUAS at the same time without an FAA waiver. Technological advancements and industry need are challenging this regulation, however. We introduce a method to safely control multiple sUAS from a single software by applying methods employed by Air Traffic Control. The software sends drones information conditionally, so the software checks where aircraft are in relation to other aircraft before sending new waypoints to help manage airspace deconfliction. We also incorporate new findings that we learned in developing and testing our methods for both drone software and hardware. The ability of our software to work with agnostic aircraft presented challenges over the course of the research which we had to overcome. From this research, we provide recommendations to help the FAA adjust the 14 CFR §107.35 regulation to allow software to safely control multiple sUAS in the National Airspace System and advance the UAS industry.

Nomenclature

ADS-B = Automatic Dependent Surveillance - Broadcast

ATC = Air Traffic Control

AGL = Above Ground Level

CCP = Chinese Communist Party

COTS = Commercial Off The Shelf

CFR = Code of Federal Regulations

ft = Feet

GCS = Ground Control Station

GHz = Gigahertz

GUI = Graphical User Interface

IMS = Integrated Management System

IoT = Internet of Things

IP = Internet Protocol

LAMP = Linux, Apache, MySQL, PHP

LiDAR = Light Detection And Ranging

MPEG = Moving Picture Experts Group

MOSA = Modular Open Systems Approach

NAS = National Airspace System

NDAA = National Defense Authorization Act

PI = Principal Investigator

R&D = Research & Development

RP = Raspberry Pi

RPIC = Remote Pilot In Command

RTB = Return To Base

SAR = Search And Rescue

SOP = Standard Operating Procedures

sUAS = Small Unmanned Aircraft System

USAF = United States Air Force

I. Introduction

The 14 CFR §107.35 regulation restricts a person from operating or acting as a remote pilot in command (RPIC) or visual observer in the operation of more than one sUAS at the same time without an FAA waiver. A common method for operation of multiple sUAS that is in practice today is the operation of swarms (multiple sUAS of the same exact model working together on a single task – e.g. pretty lights in the sky for entertainment). It is dangerous to have human pilots for each drone in these instances versus the nanosecond accuracy of computerized control. However, the software controlling swarms treats all drones as the same and pre-programs each drone with its mission derived from a flocking/swarming simulation. If a danger presents itself in the course of flight, there is no way to autonomously make an adjustment to move to safety. Additionally, due to the nature of current restrictions, applications of this

technology are limited to very small areas and each instance necessitates FAA inspection and certification. The requirement for FAA involvement for each use of drone swarms puts undue constraints on the agency and makes multiple drone operations impractical as an everyday commercial use case.

sUAS are proving that they provide utility beyond entertainment. They are increasingly being used for inspection, disaster prevention/response, surveying, and protection. They can carry out more dangerous tasks and keep humans out of harm's way. As drone technology is improving, the application of the technology is expanding. The applications, however, are currently limited by 14 CFR §107.35 and the restriction of one pilot to a single sUAS. However, we believe that technology exists to ease those restrictions and safely expand the utility of drones.

Through this project, we have tested methodologies which have evolved over the course of this period of performance to test the ability of our Harbinger™ Control platform, software control in general, and operating procedures to enable a single pilot to safely control multiple sUAS, much in the way an air traffic controller monitors and directs multiple aircraft. These new methodologies and procedures are far more conducive to commercial applications and emergency response; they represent significant savings in time, expense, personnel, and consequently lives. In one conversation with a representative from US Customs and Border Protection at Thunderstorm 24-2, an operation sponsored by the Office of the Secretary of Defense, the officer stated that due to current regulation restrictions, drones (sUAS) are minimally effective because one drone requires one pilot; greatly reducing their intelligence gathering capabilities.

The software and methodologies used in this project exceed existing levels of safety and optimize drone operations to relieve burden on the FAA while creating a standard operating procedure of sUAS across multiple commercial vectors. It incorporates ATC practices used for manned aircraft and assigns sUAS to specified altitudes and sectors. It also sends mission waypoints to drones based on conditions (i.e. other drone positions, weather, ADS-B, etc.) so that missions can be easily adjusted to accommodate safety in real-time. We also favored agnosticism, making no modifications to aircraft firmware nor hardware (aircraft remained in their COTS state as purchased). The software simply connects to existing ground control stations. This expands the use cases for sUAS and increases their utility, while greatly reducing costs and resources; effectively making a multi-drone operation more practical, safe, and effective, in even the most dynamic environments.

Interfacing with both drone software (MavLINK and variations) and hardware presented challenges, which we describe as well as the methods used to overcome these challenges. We present recommendations to enhance the

capabilities and safety of sUAS (in general) and for the FAA to advance the ability to safely enable multi-drone control from a single software system/RPIC into the National Airspace System.

II. Methodology

The methodologies employed in this project are 3-tiered: divide airspace based on task and airframe type, use real-time telemetry to adjust mission parameters on a per drone basis, and careful planning and review of successes and failures between each and every flight/test.

A. Dividing Airspace Based on Task and Airframe Type:

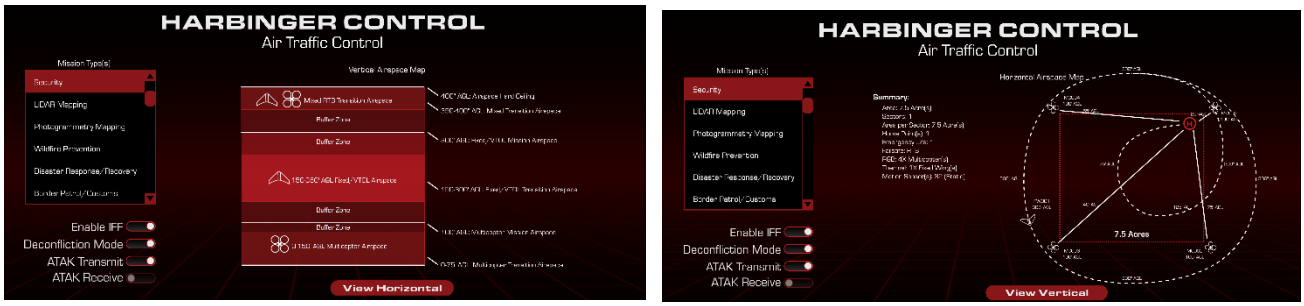

Fig. 1: The images above depict the ATC (Air Traffic Control) screens of Tempest's Harbinger™ software.

By dividing airspace both horizontally and vertically, and assigning mission and transition altitudes and sectors, multiple drones are deconflicted and free to conduct individual missions safely to contribute to an overall goal. This methodology is dubbed "Drone Battalions" by Tempest (due to the similarity in organizing drones based on capabilities being similar to structuring platoons and batallions in the military).

1. Deconfliction in Practice:

In this example, the four multicopters stationed at the 4-corners are at an AGL altitude of 75', and their mission airspace is designated vertically as 0'-150'. We add a buffer zone to separate the multicopters from the fixed-wing mission area of 150'-350' to keep the orbiting fixed-wing well away from the multicopters, and then add a transitional airspace (where sUAS use this airspace to transition from launch/landing out to mission areas). By launching the fixed-wing first, it can get into position without risk to multicopters. Then, by dividing the entire mission area horizontally into quarters, each multicopter can ascend/descend through the orbital path of the fixed wing to their

mission area only while the fixed wing does not occupy that sector. This approach has also been successfully applied to other missions and aircraft.

Tempest is pleased to state that this method of dividing airspace led to zero collisions during any test (even close-proximity multiple drones inside a relatively small, netted enclosure).

B. Use Real-Time Telemetry to Adjust Mission Parameters on a Per Drone Basis

By tracking drone locations in real-time using the Harbinger™ Control software (the pyMavLink, MavSDK, and LAMP database/configuration console) interface between the GUI and the drone GCSs to communicate with each sUAS GCS simultaneously while gathering telemetry and camera/sensor data, we are able to maintain full situational awareness over every drone's exact position, waypoint statuses, and health in real-time. This allows the software to identify potential risk-points in missions and pause drones to let another catch up and reach a safe position before continuing. Should telemetry links be broken, all drones "freeze" in place, and can safely be piloted by a human to the ground either one at a time, or simultaneously (in the case of multiple pilots present). We also identified rally points, or "ditch points", locations at which the software could land the drones (separate from their "home" point) which were created in real-time in relation to the drone's current position.

In short, real-time telemetry and conditional-based control (over the simulated algorithm/upload of entire mission) methodology has proven critical to safety and successful operation.

C. Careful Planning and Review of Successes and Failures Between Each and Every Flight/Test

We made the time to discuss and review flights after each test to learn and make improvements. This driving philosophy is far more than "go out to the field, fly, and post-mortem." Each individual flight and test (from a simple arm/disarm test to a full 6-drone mapping test) had a gathering at the command post (a trailer housing the server and software development team) wherein the primary investigator and project manager were briefed on the meaning of the telemetry readings as well as items in the code and circumstances which worked and failed, and where discussion, analysis, and further action were decided.

Seemingly slow, this methodology enabled Tempest to adjust mission parameters quickly and account for unforeseen circumstances such as signal interference, sensor discrepancies, code errors, and even differences between drones on how they parse MavLINK commands / give telemetry.

This methodology enabled a plethora of data (even unanticipated data on the state of the drone industry) which may or may not have been gathered using a "post-mortem" methodology and contributed greatly to this research. This level of detailed data and analysis is significant and necessary to inform the industry.

III.Results and Discussion

Assigning drones to specific altitudes and latitude/longitude coordinates is critical to safely control multiple drones from a single pilot, preferably a software system that supports detailed aircraft management. This was especially true in the netted enclosure where the drones were only 10 ft apart but were able to transition past each other without incident. Granted, highly controlled environments (like light shows) can (mostly) still function safely with minimal collisions using the pre-programmed "simulation" motif. But in dynamic commercial, military, responder, and safety applications, a more structured method is required.

The method of sending conditional waypoints is the other critical puzzle piece to safe flight. Algorithms in flocking and swarming can still be applied; but they must be able to be adjusted in real-time as circumstances change (i.e. SAR operations during a wildfire, and an inbound manned firebomber coming through the area – drones MUST depart immediately and then resume operations after safe again).

Over the course of developing software to safely control multiple aircraft, we have identified both challenges and methods to overcome them. These challenges were both related to software and hardware. Using more than four drones, there were additional operational challenges that were even more significant than anticipated.

D. Software

MavLINK is the primary "language" for communicating with sUAS. But it is nowhere near a "standard" for sUAS. With major differences in compatibility for commands sent, structure and timing of those commands, as well as the way telemetry and responses are sent back, it's more like considering "Earthling" the same language spoken on Earth due to the fact all our languages are voiced (with exceptions for sign language). Although there are commonalities, the differences are far more prevalent.

MavSDK and pyMavLINK are examples of advancements meant to interpolate some of those differences and serve as programable interfaces to MavLINK. However, as they are "maintained" by communities, the preferences toward certain controllers and firmwares (i.e. Ardupilot vs PX4) become obvious in their compatibility.

In testing the Control feature of Harbinger™, we learned that different aircraft and controllers behaved differently with the different controllers, firmware, and accessories; adjustments needed to be made at the MavSDK and pyMavLINK level so that the drones would behave as desired. Since few have done this work to support multi-vehicle control, there is little documentation on how to address these inconsistencies. We therefore focused much of our R&D efforts on this issue.

Ardupilot and PX4 are the two primary controller firmwares, and we noticed differences between these as well. Our Control software sends conditional commands to the drones instead of uploading a single, complete mission. This design implementation supports the ability to safely control multiple sUAS – the software checks the drone's surroundings before it gives it a new mission waypoint / command. Because our software was designed to be drone agnostic, we utilized aircraft that had both Ardupilot and PX4 controllers. We noticed that drones with PX4 controllers were able to hold in between receiving mission waypoints, whereas drones with Ardupilot controllers would drop connection if commands were not constantly sent to the controller. Ardupilot can accept new mission waypoints ONLY while it is carrying out a mission so that it can move from one mission to the next, but once a mission is completed, it cannot accept new waypoints and will have to land. As a result, when simultaneously controlling drones that use Ardupilot *and* PX4, one must account for this discrepancy and make sure the timing of the drones receiving waypoints aligns.

PX4 is not without its issues either. When sending new waypoints (for example, you've already completed 10 waypoints and this is now the 11^{th}), you must add 10 "dummy waypoints" to the PX4 mission so that it realizes this new waypoint has not yet been completed. We also noticed that increasing the number of "dummy waypoints", which occurred with longer missions, increased the time it took for the PX4 drone to receive its new waypoint. For example, during the 2-drone boustrophedonic mission, the first waypoints took less than 0.25 seconds to upload – as indicated by the length of the 'pause' between reaching a waypoint and proceeding to the next. Around the 16^{th} phase of the mission, it took nearly 25 seconds to receive the next waypoint. Eventually timeouts would happen if we didn't also increase timeout tolerance along with new waypoints.

These differences between firmware compatibility with MavLINK commands proved not only to be frustrating, but a little like playing "Where's Waldo" when trying to find documentation and compatibility. Experts on the subject matter were confined and compartmentalized into specific makes and models of drones. Although it was a frustrating

experience, it wasn't without merit. Our compatibility database and overall knowledge are far more robust as a result of this effort.

MavSDK supports multiple commands that perform the same function. This redundancy is more confusing than helpful, however, as we noticed syntax differences between Ardupilot and PX4. For example, the "loiter" command is used in Ardupilot but does not exist for PX4. Instead, PX4 uses "hold". The resultant functionality is the same, however developers should be aware of the differences when working with the two different flight controllers and trying to control multiple drones. These differences spider to nearly every single command in MavLINK, meaning that this "standard" is really not standard at all.

E. Hardware

Finding hardware that was NDAA compliant for this contract was a bit of a challenge (to say the very least). There are multiple versions of NDAA and some are more restrictive than others. So much so, that this requires a brief explanation. For example, take the Herelink (black) controller. Because a touchscreen inside is derived from a Chinese-made mobile device (keep in mind that nearly all iPhones come from the CCP), it is compatible with NDAA 2018 (the standard for this contract), however it is not compatible with 2021. The Herelink v1.1 is compatible with 2021, but not 2023, for that you must use Herelink Blue. Some companies claim NDAA compliance, but upon further investigation are truly NOT NDAA compliant. It's a veritable minefield of variables to sift through with no common listing (so much so that Tempest has considered making a "notccp.com" website to host a list for the community – and yes, we did buy the domain).

Fortunately, there are increasingly more drone manufacturers in the US working to be NDAA compliant, with some even becoming Blue certified for U.S. military use, or "Green" for use with infrastructure. But these certifications are expensive and not conducive to startups and innovation (costing > $100,000 just to **start** the Green certification process).

As we were identifying aircraft to use for our test flights, we spoke with several drone manufacturers to understand their aircraft control mechanisms to make sure it would work with our system. Amazingly, and inspiringly, nearly every drone manufacturer we interfaced with was more than just willing to help but went above and beyond expectations to enable our success in this project. As a reminder, Harbinger™ interfaces with the drone's controller and does not modify any of the hardware. As detailed above, the majority of sUAS use MavLINK and PX4 and Ardupilot controllers, so it seemed straightforward to be able to control any drone that used MavLINK through either

Ardupilot or PX4. However, we found that there were some variations that manufacturers made such that different drones behaved differently, even when using the same controller type. There were even differences between the same exact make and model of drone based on firmware version on the drone, or controller, or a combination thereof.

Much of this discrepancy may be due to manufacturers not preparing for outside control, let alone multi-vehicle control and/or their drones being used with other aircraft / control systems. For example, one drone manufacturer hard coded IP addresses into their controllers (meaning they were only able to plug their ethernet interface directly into a GCS computer – not a network switch to work as an IoT device). Since a primary method of connection uses IP addresses to identify hardware, if there are multiple pieces of hardware that all have the same IP address, the software will not be able to send information to each individual hardware. We identified a viable solution, with the help of the Chief Technology Officer of a major US drone manufacturer, by introducing a Raspberry Pi computer to act as a command / telemetry forwarder to give each drone a different IP address (via the RP having a unique ethernet address and using MavP2P, a command forwarder, to send commands to a second ethernet port attached to the GCS). Although new hardware was added, the drone itself was not modified, nor was the controller. These RP devices are considered additional network hardware.

We also identified manufacturers who adjusted the Herelink/Cube IMS functionality to "lock down" proprietary functionality and/or give compatibility to sensors or devices where none existed before (i.e. what we call in software a "kludge" to make something work that should not). This adds a layer of complexity because a drone with an Orange Cube is expected to behave like an Orange Cube. But when a drone manufacturer changes the behavior of the IMS without documenting it for users, expectations are not aligned, especially when they still advertise "MavLINK compatibility."

Because we also work with the military, we found that drones specifically made for defense have different requirements from civilian aircraft. Specifically, maintaining security of the aircraft should it fall into adversarial hands. As such, these drones may not use standard IP addresses or frequencies for drone control. For example, for our software to control the defense drone Skydio X2D/X10D, we needed to add a Microhard transceiver to send missions to the drone in addition to needing the controller. This becomes the ONLY example to date where our software needs direct control of the drone (bypassing the existing GCS controller which comes with the sUAS) – even among our Blue drone fleet. We are still working with Skydio to include the defense drones into our database, but their "MavLINK" command set is so vastly different than other drones that is barely in the same lexicon. Their civilian

versions are much easier to work with as they can be accessed across the network without bypassing the GCS, but still have the "MavLINK" issues.

Of course, with the current state of the drone industry (one pilot to one drone with the possibility for a computer to be plugged in), these methods employed by manufacturers are acceptable. However, if the industry is ever to seriously consider moving toward a more "one to many" or even a single operator but more MOSA approach, standardization of MavLINK compatibility must be addressed in any SOP. Manufacturers tout "MavLINK compatible," but the statement is so incredibly overarching as to lose all relevant meaning other than a mere buzzword. We will dive into further detail later, but a consortium (similar to the MPEG consortium our PI is a member of) to standardize MavLINK's command structures, syntax, and timings is needed so that no one company can deviate so badly as to claim compatibility when they are truly proprietary.

Finally, there seems to be little-to-no regulatory standards/airworthiness that commercial drone manufacturers have to meet. For example, when preparing flights for different military agencies, we needed to provide detailed information about each aircraft for each government agency. There is no central depository of information to know how a drone is going to behave and no certification that the drone will behave that way. As an example, when we built a drone for our testing purposes (not for commercial sale), we identified that when the IMS was unshielded, the barometer was affected by the wind and the altitude readings were skewed. When we built a windshield, the altitudes did not vary, even in mildly windy conditions. We noticed a similar problem with a commercially made drone and told the manufacturer of the problem and a possible solution. It seems that there was no testing done under windy conditions. Without regulatory standards, there may be inconsistent quality assurance testing.

This recommendation is made while bearing in mind that we have absolutely no desire to stifle the "experimental" nature of the drone industry. However, when mass produced beyond the beta testing phase, some sort of standardized air worthiness certification is critical. This air worthiness certification should also be done at cost (or even subsidized loss) to the FAA to keep innovation and the "scrappy startup" nature of the drone industry alive and well, while simultaneously increasing safety and standards.

F. The Unexpected

When we did large scale tests, we noticed unexpected issues relating to frequency and connectivity. During our test, we set up tables with pilots sitting side by side with controllers in front of them. The tables were under a large metal awning (the door of the hanger). The aircraft were separated and some aircraft were not in direct line-of-sight

with their controllers. We noticed that when the controllers were next to each other and shielded by metal (the hangar door that RPICs were using as a sunshade), some drones were unable to connect to their controllers. We attributed this problem to frequency interference and separated the controllers/pilots and moved them out from under the metal awning. This enabled the drones to connect to their controllers and thus our software. However, the further drones got from the controller, the more susceptible they were to interference.

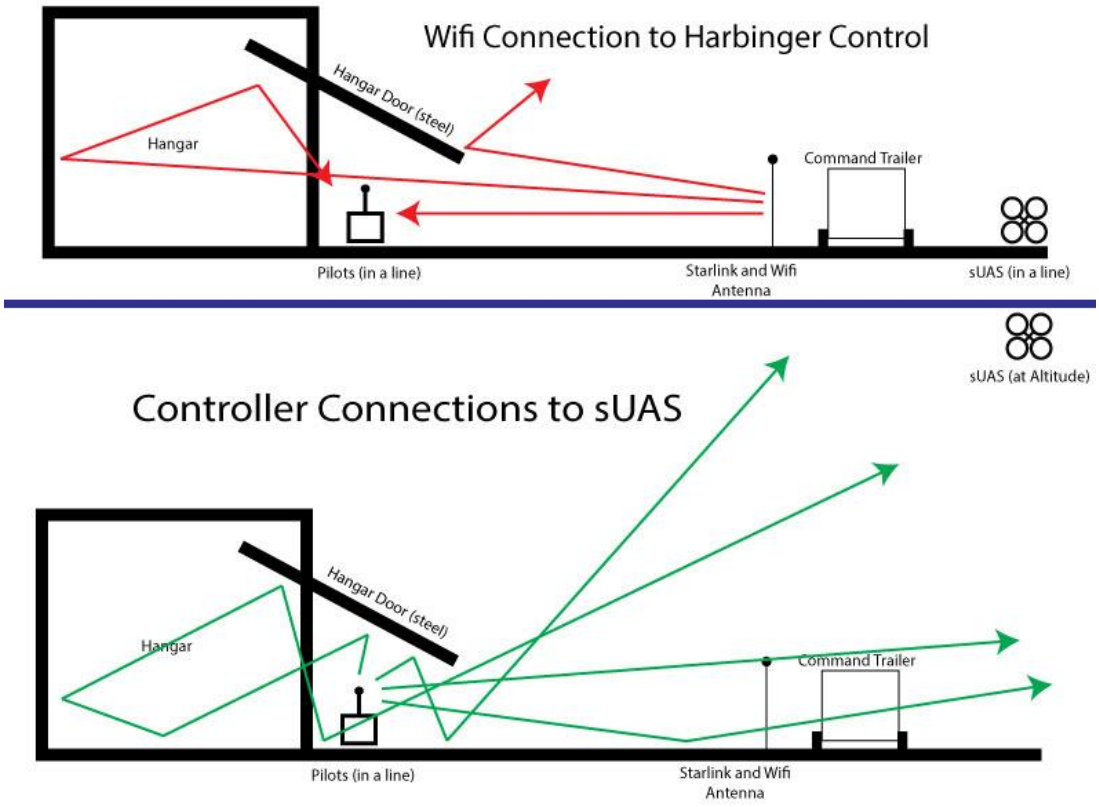

Fig. 2. The above figure depicts signals bouncing and even interfering with themselves through echoes such that signal integrity to the sUAS and the command trailer was significantly compromised

We used Starlink to provide internet connections from the software server to the controllers and from the controllers to the drones. Once the drones launched and their missions were started, we noticed that some drones lost connectivity to their controllers, which prevented the software from sending missions. We had a similar experience if the Starlink antennae was in a direct line between the drone and the controller. We hypothesized that this loss of connectivity was due to a weakened Wifi signal at greater distance combined with the energy from the Starlink satellite link.

There is a direct correlation between the number of drones, and the distance between controllers and distance from the base Wifi station. That correlation is also exponential. We suggest that no more than 4 drones be attached to one Wifi access point on the network and adding additional Wifi access points for every 4 drones. We also suggest Starlink antennas be well away from the flight area. In a later demonstration for the USAF, we flew a LiDAR mission, and we noticed the laser readings when over the Starlink antennae were skewed (we suspect from a GPS signal interference). As such, we also recommend a 15° angle minimum and 150 feet away from any flight path when using a Starlink antennae.

Fig. 3: The image on the left shows the interference when over Starlink on the LiDAR scan, while the image on the right is with Starlink powered off.

IV.Conclusion and Future Work

This research showed that it is possible for a single software system to safely control multiple sUAS in the National Airspace System. The methods we applied, dividing airspace by airframe and mission type and the delivery of conditional waypoints, proved both practical and achievable. At the outset of this project, we thought that this research would be primarily centered around those methodologies. Rather, we found the implementation of multi-drone control, from variations in the drone control software, firmware, and hardware, to issues with frequency emissions, presented much more of a challenge. We delineate these challenges and present recommendations to support a future where multiple sUAS can be controlled safely by a single software in the NAS.

Surprisingly, we found that signal interference is not taken nearly as seriously in drone management as it should be. We highly suggest further study on how signals are affected in the standard 2.4 GHz controller range by other consumer items such as Starlink and mobile devices for connectivity. We also suggest studies be done on quantity

and strength of signals with such items as mesh and airborne repeaters to aid in filtering the interference native to multiple drone operations (especially from a centralized control). Signal interference is a primary concern for safe multiple drone operations.

Tempest finds that real-time conditional-based control is critical for safe one-to-many operations. Pre-programmed flight patterns (no matter how advanced the algorithm) cannot possibly take in all variables necessary to conduct an operation (especially over populations or assets). A system must be able to adjust flight paths in real-time based on any situation that arises, along with an immediate emergency "freeze" feature (such as our Harbinger™ Control system has), should that communication encounter interruption for any reason, with clear and concise instructions and procedures delineated for such instances such as:

Should only one qualified pilot be present:

- Pilot analyzes situation to see if re-connect to server is safe
- If so, reconnect and analyze if safe to proceed with mission, RTB, or land in place / "ditch"
- If not, pilot takes control (one at a time), landing the drones in place, or at "designated ditch" points from drone with lowest battery to drone with highest charge, re-analyzing the situation as they go

Should multiple qualified pilots be present, the same procedure except with the direction of one designated "site commander."

MavLINK absolutely and critically MUST be standardized. Redundant commands must be deprecated while maintaining an open architecture for advancing the technology. We feel that an official counsel or consortium should be convened on a voluntary basis (much the same as the MPEG consortium maintains control over the standards for video compression) of industry professionals in drone design, manufacturing, software development, and from a mixture of demographics (not just the enterprise-level companies, but ONLY NDAA compliant companies should be invited). This consortium can be hosted by an organization that has 501c3 status so that it is not corrupted by proprietization native to companies for profit. Furthermore, access to the standards should remain free or subsidized to zero cost so as to enable innovation in the United States drone industry. Barriers to entry should be lowered, and never raised for those with new ideas for the industry. The consortium can also update and maintain MavLINK documentation, as well as prepare for and support advances in the industry, such as multi-drone control from a single software.

Similarly, and finally, a certification for "commercial sale ready" in the drone industry should be created. Again, this must be subsidized by the government so as to not stifle innovation while making us more competitive with the CCP. Green certification is a start, but the barrier to entry is difficult for even the larger manufacturers in the US. Again, barriers to entry for innovation should be lowered and never raised. Increased competition and innovation while also increasing safety is possible, and that is the meat of our recommendations: standardization, standardization, standardization.

The benefits of a single software controlling multiple sUAS are significant and are worth this effort and further research to advance this capability. For example, when mapping a large area, what took a single drone 22 minutes to complete was carried out in 2.5 minutes by six drones. This time savings could mean the difference between life and death when applied to a search and rescue mission or the prevention of a wildfire, for example. Multi-drone control from a single software also reduces costs and the need for additional personnel. Instead of requiring a single pilot to control one sUAS, personnel can be reallocated to other tasks important to the mission. It also maximizes the utility of the drones in information gathering, where the use of more drones contributes to more information.

The applications have already presented themselves. For example, our conversation with a representative from US Customs and Border Protection revealed that while drones are useful, they are also restrictive when used under the current regulatory environment. Requiring one pilot to a single drone drains personnel resources and greatly reduces drone utility in intelligence gathering capabilities.

The ability of a single pilot to control multiple sUAS safely and autonomously frees up other highly specialized personnel to focus on other tasks required of the job. This can be applied beyond US Customs and Border Protection, for example to the Military or first responders; anytime when having greater visibility to deliver total situational awareness is needed to maintain safety. Disaster relief, wildfire prevention, agriculture, and many other industries can benefit from the services that a fleet of sUAS has to offer. It's time for drone battalions and swarms to be widely deployed. As we have shown, there are means for safe, multi-drone control from a single system. For the US to truly become a power in the drone industry we must expedite, standardize, and subsidize the industry.

Funding Sources

This research was funded by a BAA from the FAA.

Acknowledgments

The authors acknowledge members of the Tempest Droneworx team, particularly Shannon V. Lombardo, who was instrumental in this work. We also acknowledge the support of the Texas FAA Test Site, Lone Star UAS Center of Excellence, for use of their facilities, support staff, and the ability to use CIRRUS for ADS-B data. We are grateful to the many hardware manufacturers who provided timely technical support, specifically Freefly Systems, Skydio, and Triad Drones, and Julian Oes for his in-depth knowledge of MavSDK.