IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ADVANCED COMMUNICATIONS AND NETWORKING TECHNIQUES FOR WIRELESS CONNECTED INTELLIGENT ROBOT SWARMS

Received March 6, 2020, accepted April 11, 2020, date of publication April 14, 2020, date of current version April 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2987983

MUSCOP: Mission-Based UAV Swarm Coordination Protocol

FRANCISCO FABRA[1], WILLIAN ZAMORA[2], PABLO REYES[1], JULIO A. SANGÜESA[3], CARLOS T. CALAFATE[1], JUAN-CARLOS CANO[1], (Senior Member, IEEE), AND PIETRO MANZONI[1], (Senior Member, IEEE)

[1]Department of Computer Engineering and Networks, Universitat Politècnica de València, 46022 Valencia, Spain
[2]Faculty of Computer Science, Universidad Laica Eloy Alfaro de Manabí, Manta 130802, Ecuador
[3]General Military Academy, University Center of Defense, 50090 Zaragoza, Spain

Corresponding author: Francisco Fabra (frafabco@cam.upv.es)

This work was supported in part by the Ministerio de Ciencia, Innovación y Universidades, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2018, Spain, under Grant RTI2018-096384-B-I00, and in part by the Universitat Politècnica de València (UPV) for the training of Ph.D. researchers under Grant FPI-2017-S1.

ABSTRACT Nowadays, Unmanned Aerial Vehicles (UAVs) have become the preferred, and sometimes the only support tool when facing critical scenarios such as earthquakes, search and rescue missions, and border surveillance. In these scenarios, deploying a UAV swarm instead of a single UAV can provide additional benefits when, for example, cargo carrying requirements exceed the lifting power of a single UAV, or when the deployment of several UAVs simultaneously can accelerate the accomplishment of the mission, and broaden the covered area. To this aim, in this paper we present MUSCOP, a protocol that allows multiple UAVs to perfectly coordinate their flight when performing planned missions. Experimental results show that the proposed protocol is able to achieve a high degree of swarm cohesion independently of the swarm formation adopted, and even in the presence of very lossy channels, achieving minimal synchronization delays and very low position offsets with regard to the ideal case.

INDEX TERMS Ad-hoc network, ArduSim, flight coordination, swarm, UAV.

I. INTRODUCTION

Currently, the adoption and use of Unmanned Aerial Vehicles (UAVs), also known as drones, is spreading at a fast pace. There are many fields of application where UAVs can be used, and a huge amount of possibilities for further development in the near future. Among the different use cases, there are situations where employing a swarm of UAVs can help to optimize some task through cooperation [1], or to parallelize tasks by supporting the redundancy of different sensors, or with the simultaneous usage of different types of cameras, among other scenarios.

Although there are already some solutions for the automation of UAV swarm flights [2], [3], in certain situations automatic guidance can be required. Examples of such situations may include applications for large-scale agriculture in search of pests or weeds [4], [5], wild life recordings [6], or border surveillance [7], among others. In this scenarios we propose MUSCOP, a protocol which is able to coordinate the different UAVs that make up the swarm while they follow a mission, which is typically planned beforehand. Then, the communications between UAVs should enable near-real-time responsiveness to maintain the consistency of the swarm.

The reliability of communications is a major problem in the creation of swarms, as UAV synchronization directly depends on the reliability of such communications. Also, the distance separating the different UAVs that integrate the swarm must remain consistent to avoid possible collision problems. Another problem that may be experienced by swarms is associated with the transient or long-term interruption of communication, which hinders synchronization, causing delays to the entire process, or even a reduction of the number of elements in the swarm.

In this paper we propose the MUSCOP protocol, which provides UAV coordination to maintain the desired flight formation when carrying out planned missions. MUSCOP uses a centralized approach where the master UAV synchronizes all slave UAVs each time they reach an intermediate

The associate editor coordinating the review of this manuscript and approving it for publication was Luciano Bononi.

point in the mission. Our protocol has been tested with three different design formations by following a line, a matrix, or a circle around the leader. In addition, our proposal has been validated using the ArduSim [8] simulation platform, which allows us to perform realistic experiments, validating the formations with different numbers of UAVs, and in two types of environments: ideal and lossy wireless channel.

Experimental results show that the proposed solution maintains the swarm flight formation stable, introducing positioning errors below 2 meters. This performance is achieved even when the UAV reaches the target planned speed (*10 m/s*), and with a flight delay between UAVs lower than 1 second, thus avoiding potential collisions.

The rest of this paper is organized as follows: Section II reviews the state of the art on this topic. In Section III we introduce ArduSim, the simulation tool used to implement and test the proposed protocol. Then, in Section IV, we detail the proposed MUSCOP protocol. The data source and methodology used to measure the performance and accuracy of the MUSCOP protocol are presented in section V. A comprehensive validation of the protocol is then provided in section VI. Finally, in Section VII, we present our conclusions and refer to future work.

II. RELATED WORKS

Nowadays, the use of UAV swarms attempts to play a vital role in different application scenarios. Such approach, however, requires solving several technological challenges prior to their adoption, including the development of swarm coordination protocols, selecting the most appropriate communications technology, and developing an API to control the multicopters. Below we provide an overview of relevant works centered in solving some of these challenges.

In [9] the authors propose an automatic control system for UAV swarms. Specifically, for their analysis, they use two fixed-wing aerial vehicles to maintain the cohesion of the formation. The general idea is to provide a mechanism based on radio-frequency pulses through which each UAV can detect its relative rank and orientation compared to its neighbors. To achieve this, the authors use a variant of the Frenet-Serret equations of motion for the trajectories of each UAV. Unlike our proposal, this solution does not focus on scalable UAV swarms with preplanned missions.

Lidowski *et al.* [10] developed a new coordination protocol for UAV swarms focused on search missions. This solution adds geographic routing to UAV-to-UAV communications in order to improve performance. The authors evaluate their novel protocol by simulation in only two dimensions. In [11], the authors present a swarm coordination proposal using the traditional 3G/4G communications infrastructure. For communication and coordination among the UAVs they use the Scalable Data Delivery Layer (SDDL). In [12], the authors use a swarm of UAVs to provide wireless communications on disaster-struck areas. This solution configures an autonomous agent on each UAV that is able to control them cooperatively. The proposed system achieves the goal of establishing communication between multiple ground stations (GS). The proposed architecture maintains decentralized cooperative control based on behavior to search for unknown GSs, and retransmit packets from one GS to another.

Later, in [13], the authors propose a solution based on ad-hoc networks to build the communications topology necessary to control the UAVs that make up the swarm. The solution is focused on providing connectivity and maximizing the coverage area. Then, Bekmezci *et al.* [14] provide a thorough study of the challenges of Flying Ad-Hoc Networks (FANETs), including topics such as topology changes, radio propagation model, adaptability, scalability, latency, UAV platform constraints, and bandwidth.

Another approach could be the use of Zigbee adapters to make up swarms of microdrones [15]. This solution is compatible with complex mobility patterns, but it is not well fitted to the real deployment of swarms in an open field, as the synchronization between UAVs relies upon a sensor network built in a test laboratory. The paper also presents details about the algorithm and the hardware used for implementation; they validated their solutions through real experiments using 20 UAVs. Similarly, in [16], the authors defined a virtual structure based formation controller for UAV swarms moving in the three-dimensional space. This proposal was extensively evaluated using simulation.

Zeng *et al.* [17] propose the use of cellular networks to connect a swarm of UAVs. In their work, the authors analyze the influence of the cellular network delays in the stability of the swarm, and the correlation between the network reliability and the swarm control stability. They establish the maximum network delay limit to grant stability, considering the proposed theoretical model. The model is validated through simulations, although the simulation platform is unknown, and only a flight formation of only three UAVs is simulated.

In [18] the authors use a controller based on a virtual leader structure to provide a rigid training. They use an approach where the controller cooperates in a decentralized way with the UAVs, allowing them to have a synchronization signal so that it achieves a predefined formation in the presence of a time-varying formation topology. Later, the authors of [19] used a similar approach, but instead they adopt a system having a switching interaction topology to achieve time-varying formations. The switching interaction topology consists of two parts. The first one uses a formation control solution based on two-loops, where the internal loop controller stabilizes the altitude, and the external loop controller drives the UAVs to the desired positions. The second one uses a formation control protocol using the adjacent information of each UAV, and where the formation can be time-varying. Also, they validate their approach in real scenarios using four quadrotors. Both proposals use algorithms based on Lyapunov to analyze the stability of their controllers.

Our work differs from all the previous ones since our protocol is able to maintain the flight formation of a swarm stable while following a previously planned mission, an issue not addressed in previous works available in the literature.

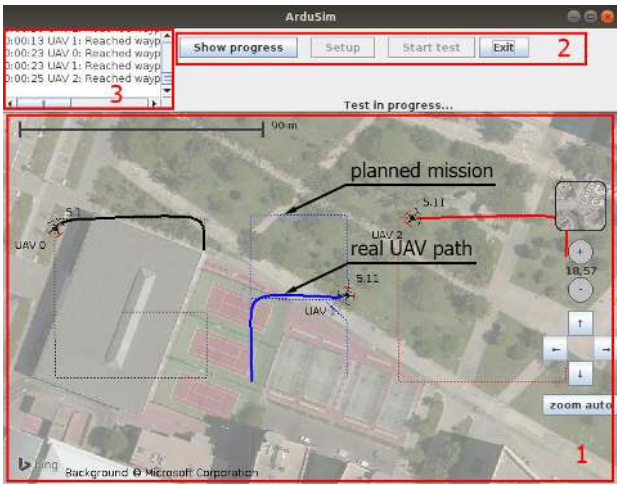


FIGURE 1. Three UAVs following independent missions on ArduSim.

The swarm leader uses the waypoints that define the mission as synchronization points to verify that all the remaining UAVs are available, and to synchronize the movement towards the next waypoint, maintaining fixed the relative location of all UAVs in the swarm. Moreover, we used ArduSim [20], a realistic multi-UAV simulator, to analyze the correctness and performance of our protocol under three different flight formation layouts: linear, matrix, and circular.

III. ArduSim SIMULATOR: AN OVERVIEW

The MUSCOP protocol has been developed and evaluated using ArduSim, a novel multi-UAV flight simulator/emulator developed by Fabra *et al.* [8], and available online [20] under the Apache License 2.0. ArduSim is able to emulate the physics of up to 256 multicopters with great accuracy, and it also simulates the communication among them through virtual Wi-Fi links.

Figure 1 shows the main window of ArduSim. Most of the window area (1) shows the movement of the virtual multicopters, and the planned mission itself (if it applies). In this example, three UAVs are drawing the letters ''GRC'' i.e., our research group initials, following a planned mission. Several buttons up on the right (2) allow to control the simulation, and help to show relevant data about each UAV in real time. Finally, the log in the upper left corner (3) shows the progress of the simulation, and messages generated by the protocol under development.

The most relevant characteristics of ArduSim are:

- **Effortless protocol deployment on real UAVs**. Ground Control Stations (GCSs) are able to communicate with real open-source multicopters using the MAVLink communications protocol [21]. ArduSim drives the virtual multicopter using this lightweight messaging protocol oriented to drones. MAVLink follows a modern hybrid publish-subscribe and point-to-point design pattern: data streams are sent/published as topics, while configuration sub-protocols such as the mission protocol or parameter protocol adopt point-to-point communications with retransmission. The protocols developed in ArduSim can be deployed in real multicopters merely by adding a Raspberry Pi with a wireless adapter to the real multicopters. A Raspberry Pi is a single-board computer with enough power to run Java applications. You only need to connect the telemetry port of the flight controller to the device (instructions available[1]), which opens a channel to communicate using the MAVLink protocol. The internal design abstracts the communication and UAV control layers, so that the implemented code works in a real multicopter the same way as in simulation, making the deployment straightforward.
- **Soft real-time simulation**. All the UAVs run in near real-time, making the protocol implementation and debugging faster than in common simulators, which often work on simulation time only.
- **Scalable**. We have proven that ArduSim runs up to 100 UAVs in near real time, and up to 256 UAV in soft real time on a high-end PC (Intel Core i7-7700, 32 GB RAM).
- **UAV-to-UAV communications**. When ArduSim is run as a simulator, the UAVs communicate among them through virtual links based on the 802.11a standard, an approach based on experiments previously performed with real multicopters. Once the protocol proves to operate reliably, it can be deployed in real devices, and ArduSim sends the application messages as User Datagram Protocol (UDP) broadcasts, thus requiring a wireless adapter connected to an ad-hoc network.
- **Complete Application Programming Interface (API)**. ArduSim provides a complete API to control the behavior of the multicopter, including take-off maneuvers, mission control, land, and so on.
- **Deployment aids**. ArduSim can be run as a simulator, and even in a real multicopter. In the second case, you can run it also as a PC Companion in a laptop, which makes it easy to start and supervise the execution of the protocol.
- **Automatic UAV collision detection**. Any swarm protocol must avoid possible collisions among UAVs. When ArduSim is run as a simulator, it provides feedback to the user whenever a collision happens between two multicopters, which helps the researcher to improve the protocol design.
- **Comprehensive experiment data logging**. When the experiment ends, either in simulation or in a real multicopter, ArduSim stores, among others, the path followed by the multicopters including coordinates, heading, speed, acceleration, distance to origin for each data recorded, as well as the same path in OMNeT++ [22], NS2 [23], and Google Earth [24] formats.

[1] https://bitbucket.org/frafabco/ardusim

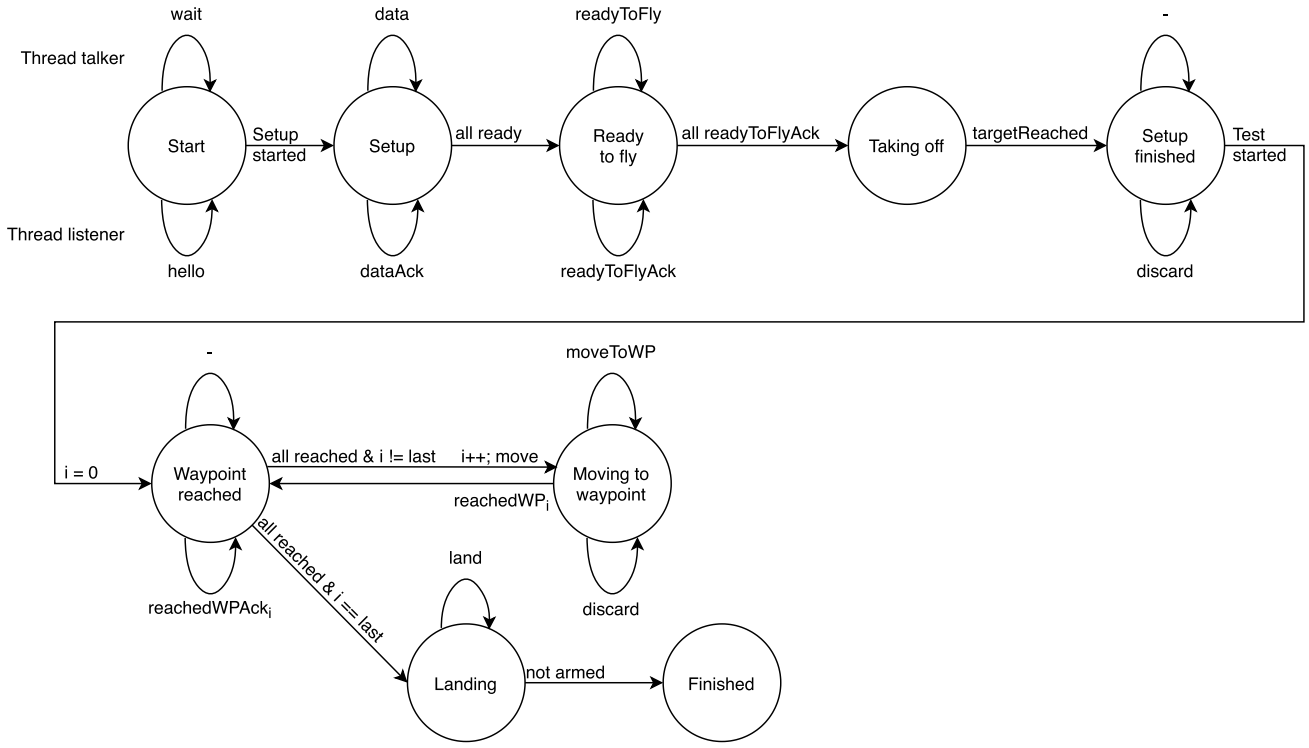


FIGURE 2. MUSCOP protocol master UAV finite state machine.

IV. MUSCOP PROTOCOL

We now present our proposed solution, which is able to coordinate a UAV swarm while it is following a mission. To this purpose we use ArduSim, which has the advantage of making developed code directly portable to real UAVs. Below we provide implementation details regarding the proposed protocol.

A. PROTOCOL OVERVIEW

The proposed protocol aims at keeping a stable flight formation while the swarm follows a previously planned mission. It is based on the master-slave model, where the swarm is synchronized by the master UAV when they reach each waypoint of the mission. Before taking off, all the slaves receive from the master a copy of the master mission with the waypoint coordinates adjusted so as to account their position offset with regard to the swarm leader in the flight formation. During flight, the UAVs move from waypoint to waypoint of their own mission, waiting on each location until the master UAV starts the next part of the mission. As the planned speed is the same for all the UAVs in the swarm, this solution keeps the flight formation steady throughout the entire flight.

The messages transmitted to coordinate the flight are handled by two threads per UAV (*Talker Thread* and *Listener Thread*). The master UAV sends coordination commands to the slaves, and the slaves send acknowledgement messages to the master UAV. The later thread implements the protocol's finite state machine, and issues the corresponding command to the flight controller as soon as the equivalent message is received from the other multicopter.

B. FINITE STATE MACHINE

The finite state machine in Figure 2 rules the behavior of the master UAV, while Figure 3 does the same for the slave UAVs of the swarm. We use circles to represent the states, curved arrows for the messages received and sent to other UAVs, and straight lines for the state transitions. Both Figures also shows messages received from other multicopters below the states (*Listener Thread*), and messages sent above each state (*Talker Thread*).

Before taking off, we assign the role of master to the multicopter closest to the center of the flight formation in order to optimize communications. As the messages are transmitted from master to slaves, and from slaves to master, we are able to minimize the distance between sender and receiver, thus minimizing the probability of message loss as well.

We now proceed to describe the behavior of the UAVs integrating a swarm. All of them begin in the *Start* state, performing a discovery procedure where the master UAV finds the available slaves. To do this, the slaves send a *hello* message to the master UAV, so it can notice their presence, and infer the number of UAVs that want to take part in the flight formation. ArduSim provides the user with feedback to help at determining when the master UAV has detected all the slaves. Then, the user switches to the *Setup* state, starting the procedure to send to all the slaves their specific mission.

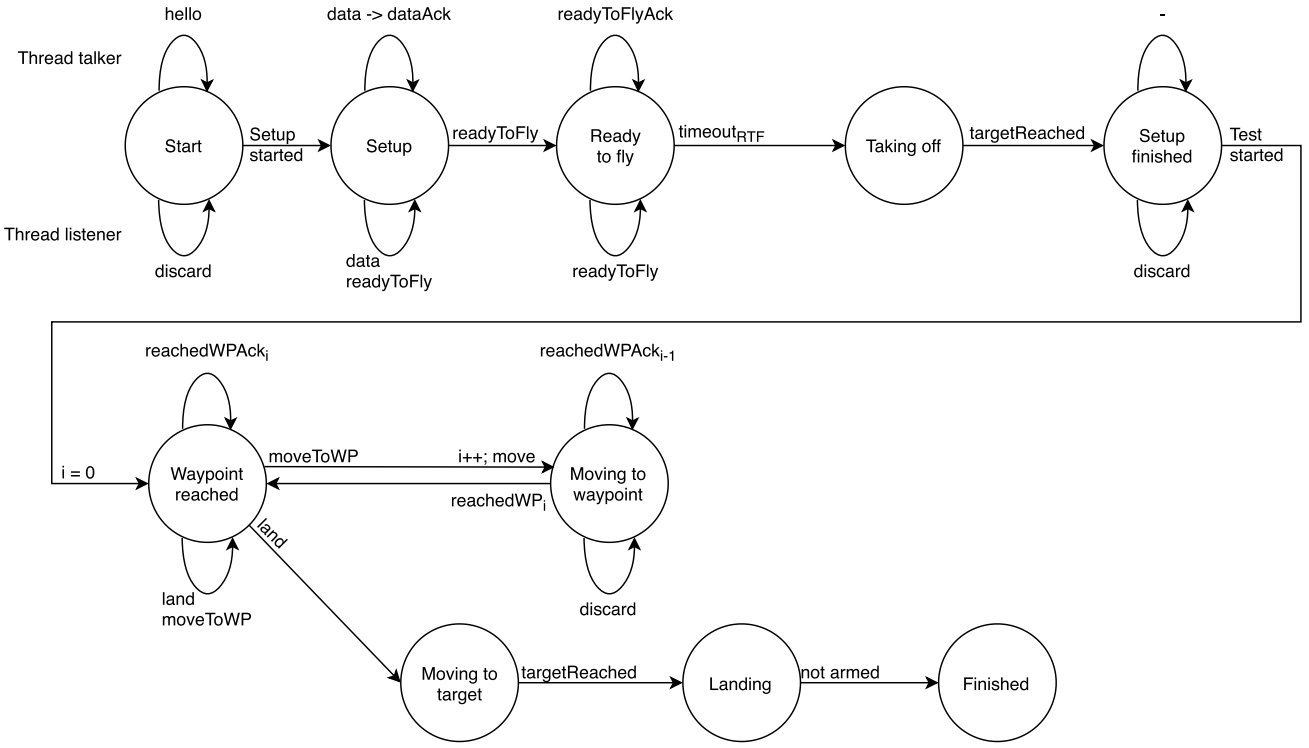


FIGURE 3. MUSCOP protocol slave UAVs finite state machine.

For this purpose, the master UAV starts by assigning all the slaves a location in the flight formation; then, it calculates the mission for each slave UAV considering their relative location in the swarm; finally, it sends them their missions (message *data*). The slaves acknowledge the reception (message *dataAck*), and only then does the master multicopter switch to the *Ready to fly* state, broadcasting the *readyToFly* message so that the slave UAVs switch to the same state, and acknowledge the reception (*readyToFlyAck* message). The master UAV begins the takeoff (*Taking off* state) when all the slaves are ready to fly, and stops sending the *readyToFly* message, which forces the slaves to also take off. The setup procedure ends when the flight formation is finally created up in the air, switching to the *Setup finished* state. We consider as our initial hypothesis that a pilot will deploy the UAVs over a flat surface, so that the fight formation is parallel to the horizontal plane, which is applicable to most of the current swarm applications. In other words, the flight formation followed by the slaves is parallel to the leader flight formation in the horizontal plane, and at the same altitude.

MUSCOP has been designed to coordinate the swarm while it follows a planned mission, and so the master starts the coordination task when the takeoff ends, and the user decides to start the flight. As observed in Figures 2 and 3, the first waypoint of the mission is the location where the UAVs are at the end of the takeoff process, meaning that the UAVs start in the *Waypoint reached* state. The master multicopter waits for all the slaves to reach that location, and only moves to the next waypoint (*Moving to waypoint* state) when it receives the *reachedWPAck* message from all the slaves. At the same time, the master UAV starts to periodically send message *moveToWP*, which forces the slaves to change to the *Moving to waypoint* state, and it also moves to the next waypoint. All UAVs keep moving until the next waypoint is reached. Periodically, the master issues the command to move (*moveToWP* message), and the slaves send back the acknowledgment of having reached the previous waypoint earlier on (*reachedWPAck* message) to increase the reliability of the protocol, as messages can be lost due to the wireless nature of the communications link. Furthermore, the transmitted messages are really short (6 and 14 bytes, respectively), which implies a very low wireless medium occupancy. When the master UAV reaches the last waypoint, it lands and sends the *land* message, including its current location. The slaves land upon receiving the message, but before that they change their location, maintaining the same flight formation while reducing the distance between them (i.e. 5 meters) in order to keep the swarm landing area small. Otherwise, they could land on unexpected places far from the master, making it difficult to collect the UAVs afterward.

C. MESSAGE FORMAT

The messages used by the master and slave UAVs are shown on Figure 4. In this section we detail their content and purpose. All the messages start by identifying the message (*type* field), and they are transmitted periodically (period of 200 ms) to increase the reliability of the protocol due

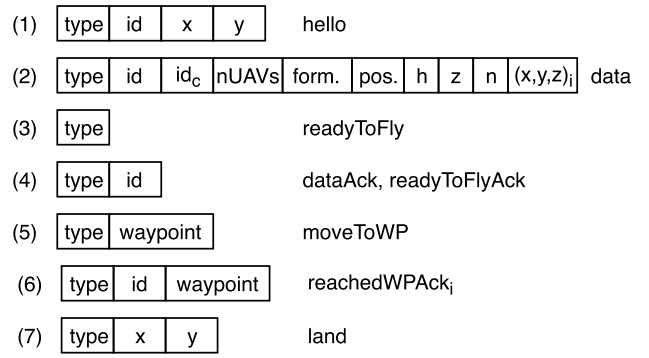


FIGURE 4. MUSCOP protocol message types.

to possible message losses through the wireless communications channel. Additionally, the messages are broadcasted with UDP to reach all the multicopters within range, minimizing the network overhead.

The slaves send the *hello* message (1) to the master right from the beginning. This way, the master UAV detects and adds them to the swarm. The *id* field contains a unique identifier for the sender multicopter, and we include it in all the messages that require identifying the sender or the receiver. This identifier is based on the MAC address of the wireless adapter on the real multicopter, while we use a unique number directly obtained from ArduSim when we run it as a simulator. The last two fields represent the current coordinates of the UAV, which enables the master UAV to define the mission this specific slave has to follow.

The master UAV calculates the relative location of the slaves with regard to itself in the predefined flight formation before taking off. At the same time, it computes the mission that all the UAVs will follow. The multicopter in the center of the flight formation will become the master UAV, and will use the original mission, while the rest of the UAVs will follow a mission that maintains fixed the relative location to the remaining UAVs in the swarm flight formation throughout the flight. We set the UAV in the center of the flight formation as the master to improve communications, minimizing the distance between the master and the furthest slave. The master UAV sends the *data* message (2) to each slave with the mission to follow:

- *id*. Identifier of the source or target UAV.
- id_c. Identifier of the UAV that will be in the center of the flight formation. When id_c matches *id*, the target multicopter will become the master UAV during the flight, sending coordination messages to the remaining UAVs. Otherwise, the multicopter will become a slave, and will accept commands from the UAV with this identifier.
- *nUAVs*. Number of UAVs in the flight formation. The master UAV requires this value to know when to move to the next waypoint.
- *form*. Type of formation, selected between the options provided by ArduSim: linear, matrix, and circular among others.
- *pos*. Position of the target UAV in the flight formation. Beside the previous two fields, this one allows each UAV to know its relative location in the flight formation.
- *h*. Heading of the UAV swarm, fixed during the whole flight. Before landing, the multicopters surrounding the center UAV approach it, conforming a more compact version of the flight formation in order to land in a reduced area, making it easier to collect the multicopters afterward. This field is needed for each UAV to calculate its location in the landing formation.
- *z*. Altitude over the ground for the take-off step.
- *n*. Size of the mission included in the message (number of waypoints).
- $(x,y,z)_i$. Coordinates for all the waypoints included in the message. The maximum number of waypoints that can fit in the message is 58, considering the maximum payload of an UDP datagram over standard Ethernet.

Message *readyToFly* (3) prepares the slaves to take off when all of them acknowledge the reception of the *data* message.

The slave UAVs use the messages *dataAck* and *readyToFlyAck* (4) to inform the master that messages *data* and *readyToFly*, respectively, have been received. The first one notifies the reception of the UAV mission, and the second the fact that the multicopter is ready to take off. The identifier of the sender is included to help the master UAV determine when all the slaves have the required data.

The master UAV synchronizes the swarm, and moves to the next waypoint using the *moveToWP* message (5). Previously, it has to wait until all the slaves reach the waypoint it is waiting at, and sends the $reachedWPAck_i$ message (6).

Finally, the master UAV sends the *land* command (7) when it reaches the last waypoint. It includes its location to allow the slaves to compute where to land, reducing the distance to other UAVs, but maintaining the same flight formation.

D. SWARM FORMATIONS

ArduSim integrates tools that allow us to easily define and use different flight formations on a swarm:

- **Linear.** The multicopters conform a line perpendicular to the orientation of the master, being the latter located at the middle of the flight formation. The distance between contiguous multicopters remains constant.
- **Matrix.** The multicopters are arranged according to a square matrix. Again, the master multicopter is located in the center of the swarm, and the distance between contiguous multicopters is the same.
- **Circular.** The master multicopter is in the center of the circle, and the slaves surround it. The slave-master and slave-slave distances can be defined by the user for this formation.

It is noteworthy that the main flight formation parameter is the minimum distance between contiguous multicopters, as it is directly related to the probability of collisions during flight.

FIGURE 5. Flight formations: i) 9 UAVs matrix, ii) 5 UAVs line, and iii) 9 UAVs circle.

Figure 5 illustrates these three different formations. As stated earlier, the UAV located at the center of the flight formation is chosen as master to improve communications performance.

Notice that the chosen patterns provide different trade-offs between communication link reliability and area coverage. The linear scenario can provide the greatest coverage area, but with a greater distance between the master UAV and the slaves (worst-case approach in terms of reliability), given a same number of UAVs, and for a same distance between contiguous UAVs. In particular, the multicopters located on edge positions will be quite far away from the master, so the messages transmitted between slave and master are prone to be lost more often, thereby having a negative effect in terms of swarm coordination times. On the contrary, the matrix formation is more compact (worst area coverage by the UAVs involved), but the distance between master and slaves is lower, which implies fewer data packet losses. Finally, the circular formation is in-between the previous two formations regarding master-slaves distance. Additionally, this distance is the same in all cases, meaning that a similar packet loss level is expected.

V. DATA SOURCES AND ERROR ASSESSMENT

We now proceed to validate the solution detailed in the previous section. First, we will include details about the used data sources, and then we will explain the procedure adopted to calculate the UAV location errors gathered during the experiments.

A. DATA SOURCE

The proposed solution requires a planned mission that the swarm will follow throughout the flight. In order to analyze the influence of the mission complexity in the flight formation error, we defined several missions by increasing the number of waypoints. The mission length was the same in all the missions (1840 m), and the swarm moves from south to north, and from west to east in alternating mission segments. Figure 6 shows three mission examples with two (brown), six (white), and thirty (green) waypoints, respectively.

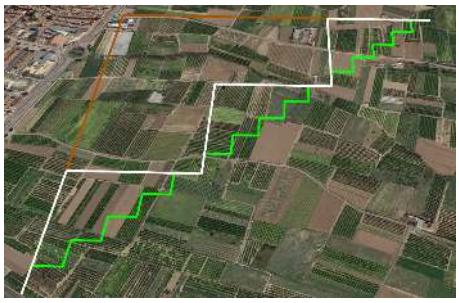


FIGURE 6. Samples of test missions with 2, 6, and 30 waypoints.

Using the previously defined missions as input, we performed an extensive set of experiments. At the end of each simulation, ArduSim provides a large amount of data, including speed, acceleration, coordinates, altitude, and heading, at different time instants along the simulation. We consider as origin of the experiment the location of each UAV recorded at the beginning of the flight. Then, we perform an interpolation of the simulated data at fixed time steps in order to get the location of each UAV throughout the experiment.

We now describe the process used to obtain the flight formation error for each of the UAVs of the swarm during the flight. Since the flight is coordinated by the master UAV, the main metric is the time delay/offset between the master multicopter and the slaves for the three formations studied: linear, matrix and circular. We can get this value given the distance offset error for each time step.

Figure 7 shows the distance offset error for a matrix flight formation. We define the real location of the master UAV (red UAV) as reference to build around it the theoretical location of the slaves (blue ''X''), given the target location (matrix in the example). We define as UAV distance offset the distance between the master and the slaves' theoretical locations. The distance offset error of each slave is defined as the distance between the expected location (blue ''X''), and the actual location of the multicopter (black UAV). The black arrows show the movement direction of all the multicopters that make up the swarm. It is noteworthy that all the UAVs move in the same direction along straight lines, and thus the offset for the entire journey will in general be parallel to the mission segments. Below we provide more details about the procedure adopted for the calculation of the position and time offsets for the UAVs in the swarm.

B. ERROR ANALYSIS

As stated before, the distance offset error (ε) refers to the distance between the theoretical location of all the slave UAVs, and their actual location, where the individual positions in the formation are defined based on the current location of the master UAV $\overrightarrow{P_{M_i}}$, and taking as reference a constant heading

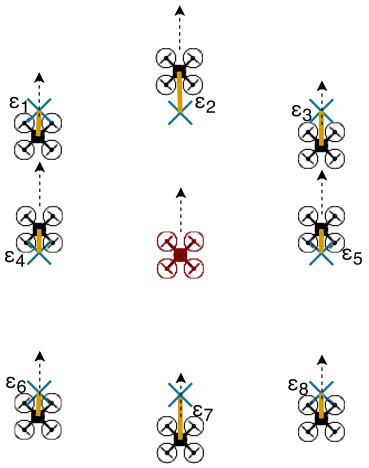


FIGURE 7. Distance offset error for the matrix formation.

for the formation.

$$\overrightarrow{P_{M_i}} = (P_{x_i}, P_{y_i}) \tag{1}$$

In equation 2, i represents the current time step for a real position on x and y axes, respectively. $\overrightarrow{P_{k_i}}$ represents the theoretical position of the slave UAV k calculated for time instant i, using the master actual location $\overrightarrow{P_{M_i}}$ as reference, and adding a calculated offset $\overrightarrow{\Delta_k}$.

$$\overrightarrow{P_{k_i}} = \overrightarrow{P_{M_i}} + \overrightarrow{\Delta_k} \tag{2}$$

where

$$\begin{aligned}
\overrightarrow{P_{k_i}} &= (x_{k_i}, y_{k_i}) \\
\overrightarrow{\Delta_k} &= (\Delta x_k, \Delta y_k) \\
x_{k_i} &= P_{x_i} + \Delta x_k \\
y_{k_i} &= P_{y_i} + \Delta y_k \\
\Delta x_k &= offset_{x_k} \cdot \cos(h) + offset_{y_k} \cdot \sin(h) \\
\Delta y_k &= offset_{y_k} \cdot \cos(h) - offset_{x_k} \cdot \sin(h)
\end{aligned}$$

As the heading h of the flight formation, and the relative offset of a slave k with respect to the master UAV, remain constant to achieve swarm cohesion, the calculated offset $\overrightarrow{\Delta_k}$ also remains constant over time.

For the time step i, the interpolated data set provides the actual location of a multicopter $\overrightarrow{P'_{k_i}} = (x'_{k_i}, y'_{k_i})$, where each value represents the location in the x and y axes.

The equation to calculate the error or distance offset of the slaves with regard to the master UAV is defined as:

$$\varepsilon_{k_i} = \sqrt{\left(x_{k_i} - x'_{k_i}\right)^2 + \left(y_{k_i} - x'_{k_i}\right)^2} \tag{3}$$

Finally, we calculate the time offset as $\varepsilon_{k_i}/v_{k_i}$, where v_{k_i} is the current speed of UAV k.

VI. VALIDATION OF THE PROPOSED SOLUTION

Up to now, we have described the data used and produced during the experiment, and the error calculation strategy. In this section, we validate the proposed solution based on that strategy. To this aim, we performed a large number of experiments modifying the number of UAVs, the distance between them, and the flight formation. The experiments were divided in three groups: (i) impact of MUSCOP and mission complexity on flying time; (ii) impact of channel losses on swarm cohesion for the given formation; (iii) impact of inter-UAV distance in swarm stability; and finally (iv) scalability analysis. Please notice that the analysis was performed while the swarm is following the mission, discarding the data gathered while the multicopters were taking off and landing. In an illustrative video[2] we run three experiments with different flight formations on the ArduSim simulator. Below we detail the results obtained.

A. IMPACT OF MUSCOP AND MISSION COMPLEXITY

In this part, our aim is to analyze the time and distance offset error due to the nature of the proposed solution, and the performance that it can achieve using an ideal lossless communications channel. During the experiments, we set a planned speed of *10 m/s*, used a linear flight formation of 9 UAVs, and changed the mission complexity while keeping a constant mission length of 1840 meters.

1) TIME SERIES ANALYSIS

In this first evaluation, we analyze the behavior of a single simulation using nine UAVs and a linear formation. The target flight speed is set to 10 m/s, and we adopt a distance between UAVs of 50 meters (our default value). The total number of waypoints in the mission is just 2.

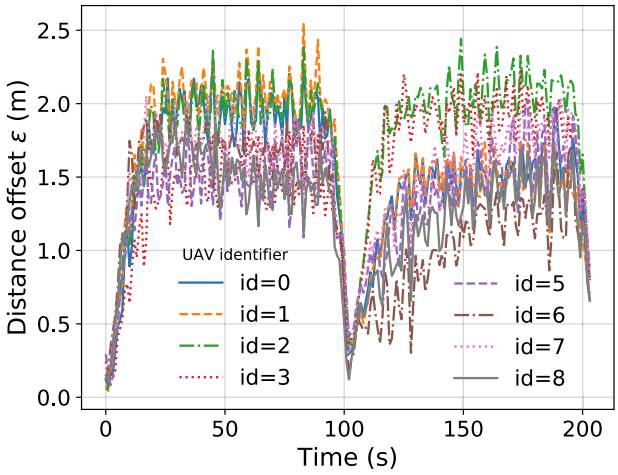
Figure 8 shows the formation error measured as position offset, and the corresponding delay offset as a function of time for a single experiment. At time 105 seconds it shows the offset between slaves and master UAV assessed when the flight coordination is taking place, when the UAV switches from the first to the second waypoint of the mission. Note that the distance offset error in the swarm formation remains low despite the time offset increases for that point. This is due to the deceleration of all UAVs, and the synchronization time overhead when arriving at this coordination point.

In general, Figure 8 (a) shows that the average error obtained for this type of formation is lower than 2 m. Concerning the time offset, Figure 8 (b) shows that the average error is of 0.16s, which coarsely corresponds to the synchronization time requirements of our protocol ($\sim$ 200 ms).

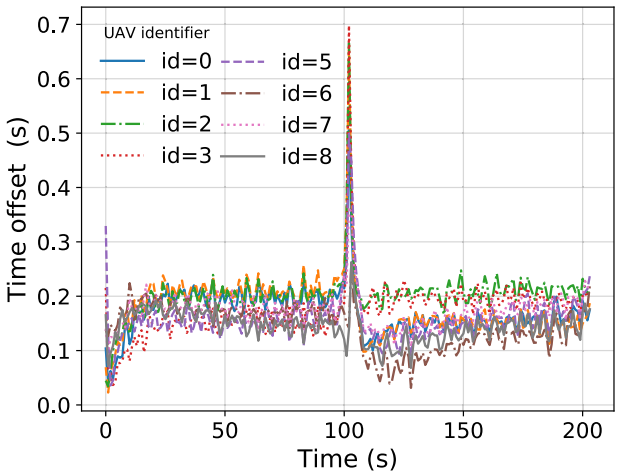
2) IMPACT OF MISSION COMPLEXITY

Once the behavior for a single simulation was analyzed, we proceeded to perform multiple simulations to evaluate the behavior when increasing the mission complexity. We define the mission complexity as the number of waypoints included on the mission, given a constant mission length; in particular, it is the frequency with which the UAVs have to change course over time, as explained in section V-A. For our tests we

[2]https://youtu.be/VLMsbL5B6tA

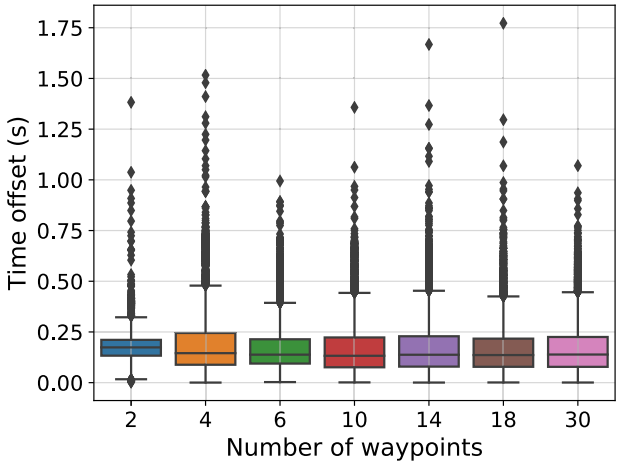


(a) Formation distance offset. (ε).

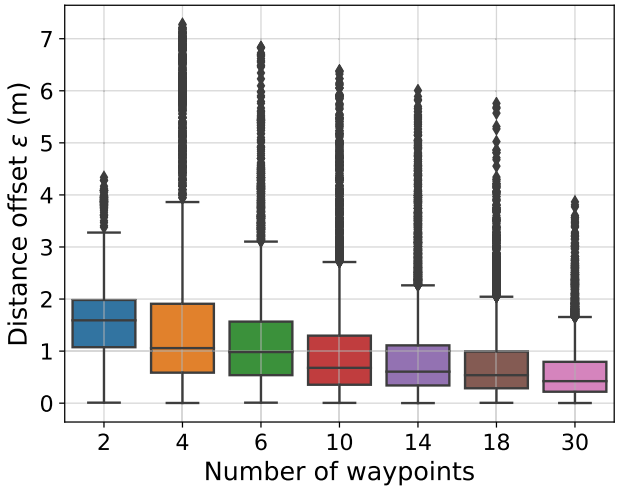


(b) Formation time offset.

FIGURE 8. Evaluation using the linear formation with 9 UAVs (ideal channel).



(a) Time offset.



(b) Distance offset.

FIGURE 9. UAVs error for 9 UAVs linear formation (ideal channel).

defined missions having 2, 4, 6, 10, 14, 18, and 30 waypoints, respectively. In total, five simulations were made for each mission type, and the speed and the separation between UAV was kept constant, adopting the values defined earlier.

Figure 9 shows the results obtained for different mission complexity levels. In general, Figure 9a shows that the time offset maintains an approximate average value of 0.20s. Also, maximum atypical values are shown, which vary according to the mission. Concerning the distance offset error, it becomes evident that, in general, it tends to decrease as the mission becomes more complex. Such behavior is shown in Figure 9b. To understand the reasons underlying this phenomenon, notice that distances between the coordination points decrease as the mission complexity increases, meaning there is less time for UAVs to accelerate and reach maximum speed values. This way, given the same time offset error, the relative slave-master error becomes smaller.

Table 1 shows the average, maximum, and standard deviation values for each flight formation evaluated. In particular, it shows that the matrix formation appears to achieve slightly better performance than the other formations, but the overall results are equivalent, as the distance between UAVs does not affect the communications link quality (ideal channel). We found that the mission complexity only affects the distance offset error, under an ideal channel, if the multicopters do not have enough time to reach the maximum speed before getting the next waypoint. On the other hand, we observe that the time offset is not affected at all, as the data packets always arrive to destination. Furthermore, table 1 also shows that the average time offset is almost equivalent to the message broadcast period (200 ms) used in MUSCOP.

TABLE 1. Overall simulation statistics (ideal channel).

Formation	Distance offset ε (m)			Time offset (s)		
	Max.	Mean	Std.	Max.	Mean	Std.
Linear	7.2788	1.0865	0.9077	1.7724	0.1749	0.1278
Matrix	6.6886	1.0496	0.8564	1.3376	0.1720	0.1209
Circular	8.2147	1.1624	0.9314	1.6323	0.1854	0.1266

TABLE 2. Flight time overhead.

Waypoints n	Reference mission (s)	Linear formation (s)	Δt (s)
2	205.33	206.33	1.00
4	226.00	228.33	2.33
6	247.67	250.00	2.33
10	288.00	293.00	5.00
14	326.00	334.33	8.33
18	364.33	374.00	9.67
30	466.00	479.33	13.33

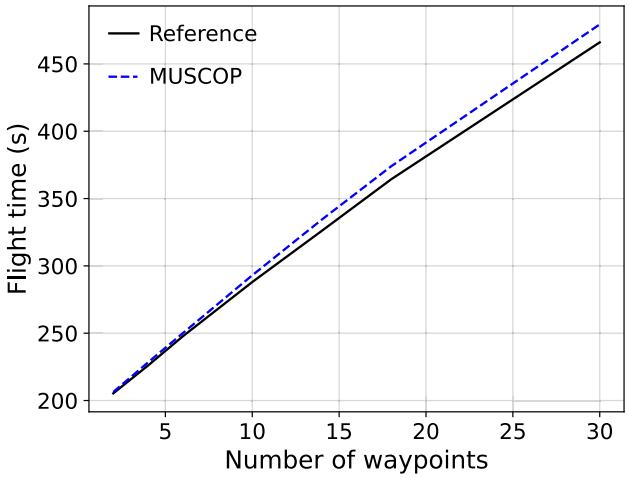


FIGURE 10. Flight time overhead using the linear formation with 9 UAVs (ideal channel).

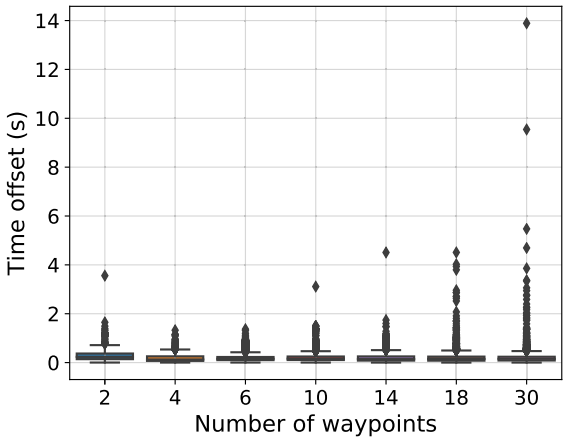
3) MUSCOP TIME OVERHEAD

The mere act of using MUSCOP may introduce some delay during the flight, as the swarm must be synchronized each time it reaches a waypoint. In this section, we check to what extent is the total flight time affected by our protocol. Table 2 shows the mean total flight time for five experiments with a swarm of 9 UAVs, and for different mission complexity levels. To determine the total flight time overhead Δt, we compared the results with the time required by a single multicopter to follow the same mission automatically (*Reference mission*), without using MUSCOP. We found that the total flight time is only increased by about 0.55 seconds per waypoint of the mission, a very low value.

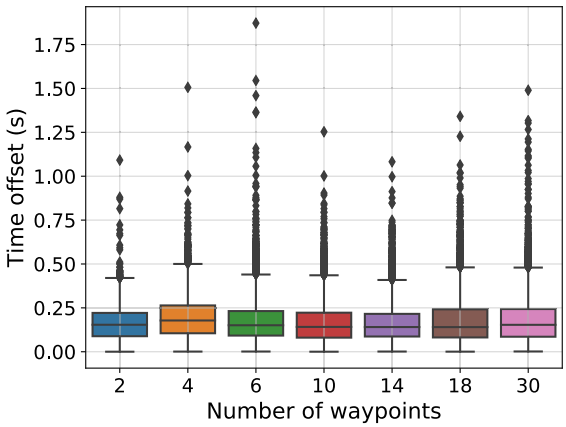
Additionally, Figure 10 confirms that the total flight time is not specially affected by the protocol, and also that the total flight time increases linearly as we increase the mission complexity, something unrelated to MUSCOP, and that is defined by the user of the swarm.

B. IMPACT OF CHANNEL LOSSES

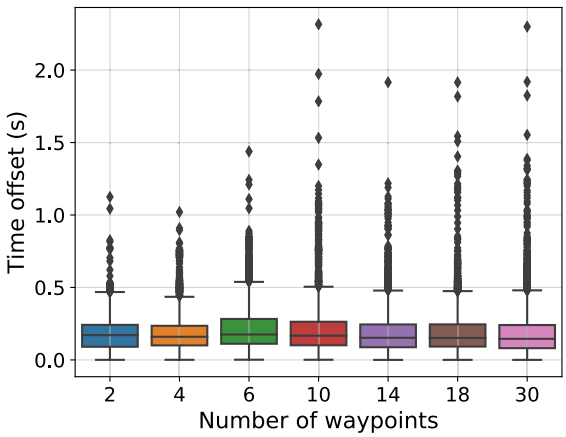
Up to now, we have analyzed the performance of MUSCOP under ideal conditions, and the impact of the mission complexity and the protocol time overhead. In this section we consider a more realistic scenario, using a communications model based on IEEE 802.11a technology, with a maximum usable distance between sender and receiver of about 1300 meters. This model was defined from the results gathered with real multicopters [25]. During these experiments, we kept the same conditions of the previous section to measure the performance using a lossy channel. Again, 9 UAVs made up the swarm, the minimum distance between contiguous UAVs was set to 50 meters, and the maximum speed was 10 m/s. Figures 11 and 12 show the time and



(a) Linear formation.



(b) Matrix formation.



(c) Circular formation.

FIGURE 11. Time offset on a swarm of 9 UAVs (lossy channel).

TABLE 3. Overall simulation statistics (lossy channel).

Formation	Distance offset ε (m)			Time offset (s)		
	Max.	Mean	Std.	Max.	Mean	Std.
Linear	13.223	1.2766	1.1753	13.8883	0.2020	0.1882
Matrix	6.9097	1.0807	0.8625	1.8721	0.1763	0.2667
Circular	9.982	1.1804	0.9920	2.3161	0.1889	0.1370

distance offset error for each of the flight formations evaluated.

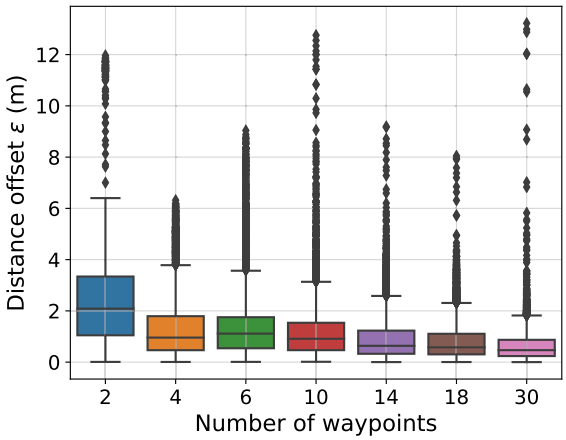
Figure 11 shows that, in general, the time offset does not vary significantly when comparing the different formations. However, compared to the results obtained under ideal channel conditions, we can now find outlier values for the delay which are quite higher than for the previous experiments. Besides, these values increase as the complexity of the mission also increases. Specifically, in the linear formation, we can see delay values that are much higher than for the rest of the formations. This occurs because, with 9 UAVs, the maximum distance from the farthest UAV to the leader is of about 200 meters.

Figure 12 shows that the average error in distance, measured as position offset differences, tends to decrease as the complexity of the mission increases. Specifically, the matrix formation introduces lower errors with respect to the linear and circular formations. In the case of linear training, we can see that it is associated with higher errors; this is due to the higher distances regarding inter-UAV communications, and whose details will be analyzed later.

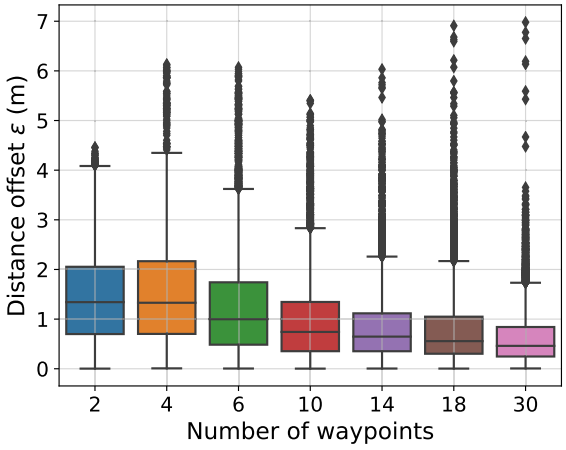
Table 3 shows the mean, maximum, and standard deviation values for each of the evaluated formations under a lossy channel. We can see that, compared to the ideal channel conditions presented earlier, it shows a slight increase in all its values. Figure 13 shows the behavior of MUSCOP under both ideal and lossy channel conditions for the three flight formations tested, considering the values obtained for missions with different complexity (with 2, 4, 6, 10, 14, 18, and 30 waypoints). In general, it becomes evident that the time delay with the ideal and lossy channels is similar, proving that MUSCOP is able to perform well enough under realistic conditions.

C. IMPACT OF VARYING THE INTER-UAV DISTANCES

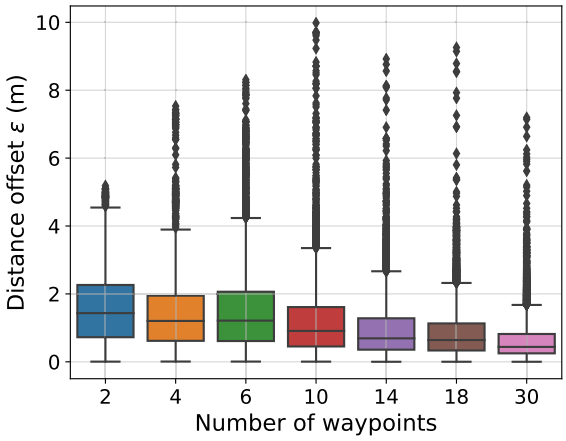
In the previous section, we found that the distance between UAVs affects the flight formation error, both in terms of distance and time offset, when using a lossy communications channel. In this section we provide details for experiments made with a linear flight formation in order to provide insight into the effect of channel losses on the swarm formation stability. For these experiments we used only three UAVs, where the master is in the middle of the flight formation. This way, the distance of any slave to the master is always the same. Then, we modified the master-slave distance from 100 to 1000 meters. The mission complexity was kept constant (14 waypoints), and we ran 5 simulations for each distance studied.



(a) Linear formation.



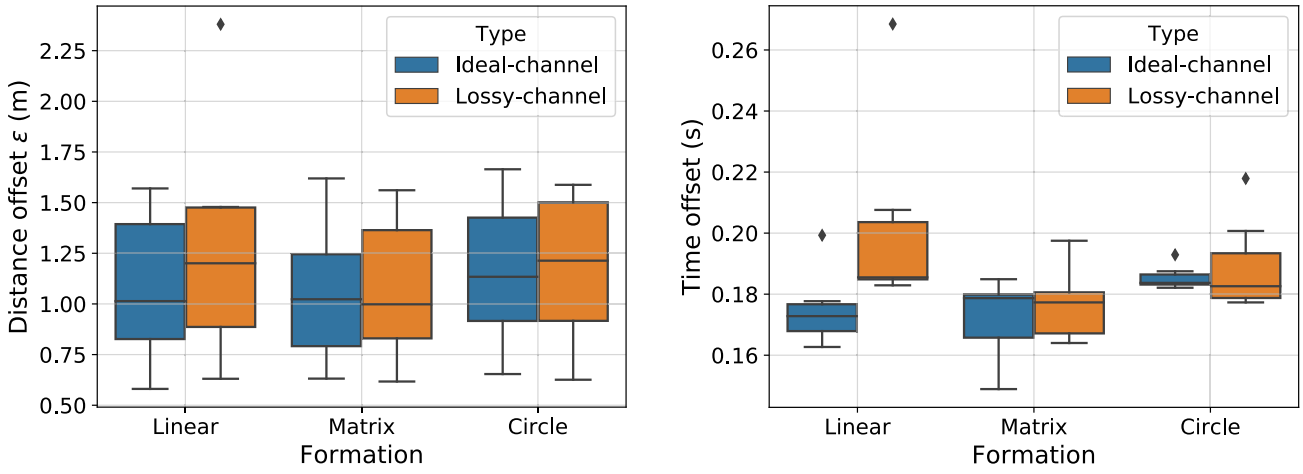
(b) Matrix formation.



(c) Circular formation.

FIGURE 12. Distance offset on a swarm of 9 UAVs (lossy channel).

In general, in Figure 14 we observe that the mean time offset grows when the distance between sender and receiver also increases. This is the expected behavior, as greater distances between them will cause packet loss, making difficult to synchronize the swarm, which could became a critical problem under realistic conditions (loss channel). We also found that



(a) Formation distance offset (ε).

(b) Formation time offset.

FIGURE 13. Performance comparison between ideal and lossy channel conditions for different formations.

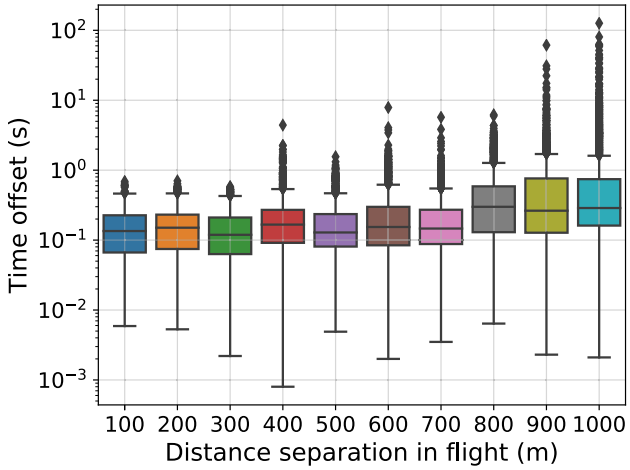


FIGURE 14. Time offset for the linear formation for different master-slave distances.

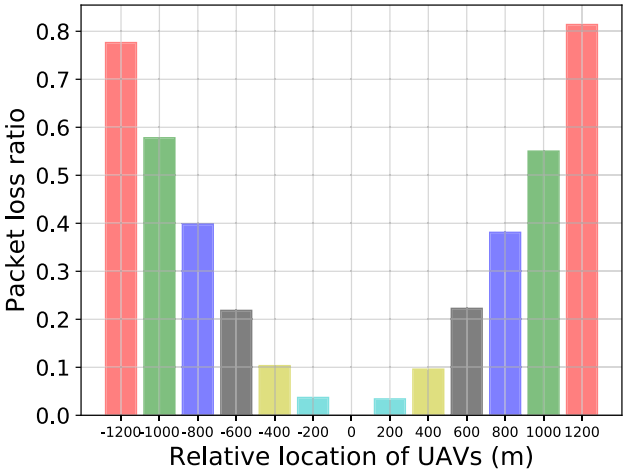


FIGURE 15. Packet loss ratio values at different distances.

the time offset is below 1 second while the master-slave distance is below 300 meters, which is a quite acceptable value.

1) PACKET LOSS RATIO ASSESSMENT

In the previous evaluation, it was observed that, at long distances, a greater time offset is introduced. Next, we proceed to evaluate the message loss ratio that occurs when using our protocol in the context of a larger swarm. For this purpose, a linear formation with 13 UAVs was used in such a way that each UAV is separated from its neighbors by a distance of 200 meters; hence, the UAVs located at the edges are 1200 meters away from the leader UAV. Several simulations were made, and the mean values were taken. The total number of waypoints for the mission was again equal to 14.

Figure 15 shows the packet loss ratio measured at different distances. In general, we can see that the UAVs that are at a greater distance have higher loss levels, as expected, a ratio that even exceeds 80% for the UAVs located in the periphery. These results evidence that ArduSim induces a very realistic model for UAV communications, and that MUSCOP is able to cope with high loss levels.

D. SCALABILITY ANALYSIS

The last set of experiments was designed to analyze the influence of the number of multicopters on performance, while also considering the complexity of the mission. To this aim, we performed experiments with the matrix flight formation, and with different number of UAVs: 9, 25, 49, and 81. For each of those swarm sizes, we carried out experiments with missions with 2, 6, 10, 14, 18, and 30 waypoints, and the same length. We maintained the distance between the leader and the furthest away multicopter constant throughout all the experiments so as to maintain the similarity among them.

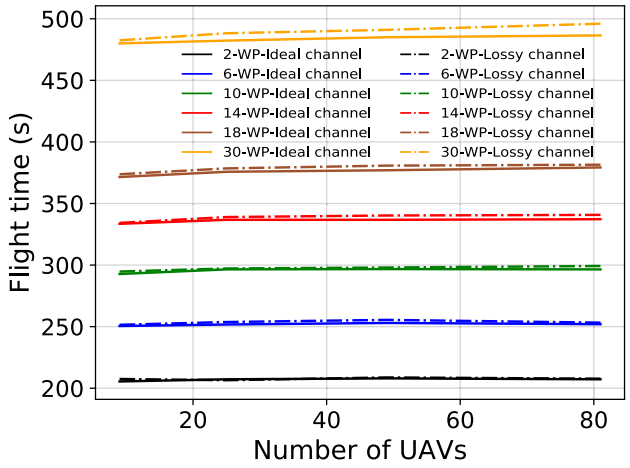


FIGURE 16. Scalability: total flight time vs. number of multicopters.

Figure 16 shows the results when we increase the number of UAVs, for missions of different complexity, and for both ideal and lossy wireless channels. In general, the total flight time is greatly affected by the mission complexity (number of waypoints), an issue not directly related to the proposed protocol. In particular, we found that the total flight time increases from 200 to beyond 450 seconds when we increase the number of waypoints from 2 to 30. It is also observed that the swarm size does not significantly affect the total flight time, which means that MUSCOP scales well by efficiently synchronizing the entire swarm. In addition, we can also see that the differences between both channel types is minimal, which means that our proposed protocol has a high resilience to packet loss.

To gain further insight on how mission complexity affects the overall flight time, we analyzed the resulting mobility pattern in more detail, finding that more complex missions introduce more acceleration and deceleration events, which causes the average flight speed to become lower, thereby increasing the total time associated to a mission.

VII. CONCLUSION AND FUTURE WORK

As the adoption of UAVs continues to grow at a steady pace, so does the complexity of the solutions targeted by these devices. In this paper we focus on applications that require the use of UAV swarms to undertake some preplanned missions. To this aim we proposed MUSCOP, a novel protocol able to adequately synchronize all UAVs in a swarm throughout all the steps involved in the flight.

Experimental validation using the ArduSim simulation platform has shown that MUSCOP is effective at maintaining the swarm cohesion for the different formations tested, and under different experimental conditions, being highly resilient to channel losses, and able to seamlessly scale to a large number of UAVs without a significant performance penalty. In fact, tests evidenced that the complexity of the mission is the main parameter affecting the overall flight times, a factor that is independent of the number of UAVs involved. Nonetheless, the flight time only increases by 0.55 seconds due to the required synchronization on each waypoint when testing with an ideal communications channel, being only slightly higher in the presence of channel losses.

In the near future we plan to confirm the results obtained by testing MUSCOP in a testbed with real multicopters, as well as to design a new protocol to optimize swarm takeoff that is both efficient and avoids possible collisions.

REFERENCES

[1] T. Miki, P. Khrapchenkov, and K. Hori, "UAV/UGV autonomous cooperation: UAV assists UGV to climb a cliff by attaching a tether," *Cooperat. Distrib. Robot Syst.*, May 2019. [Online]. Available: http://arxiv.org/abs/1903.04898

[2] H. Hildmann, E. Kovacs, F. Saffre, and A. F. Isakovic, "Nature-inspired drone swarming for real-time aerial data-collection under dynamic operational constraints," *Drones*, vol. 3, no. 3, p. 71, 2019. [Online]. Available: https://www.mdpi.com/2504-446X/3/3/71

[3] H. Shiri, J. Park, and M. Bennis, "Massive autonomous UAV path planning: A neural network based mean-field game theoretic approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

[4] B. S. Faical, G. Pessin, G. P. R. Filho, A. C. P. L. F. Carvalho, G. Furquim, and J. Ueyama, "Fine-tuning of UAV control rules for spraying pesticides on crop fields," in *Proc. IEEE 26th Int. Conf. Tools with Artif. Intell.*, Nov. 2014, pp. 527–533.

[5] D. Albani, J. IJsselmuiden, R. Haken, and V. Trianni, "Monitoring and mapping with robot swarms for agricultural applications," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–6.

[6] K. Anderson and K. J. Gaston, "Lightweight unmanned aerial vehicles will revolutionize spatial ecology," *Frontiers Ecol. Environ.*, vol. 11, no. 3, pp. 138–146, Apr. 2013.

[7] European Commission. *Autonomous Swarm of Heterogeneous Robots for BORDER Surveillance*. Accessed: Jan. 30, 2019. [Online]. Available: https://cordis.europa.eu/project/rcn/209949/factsheet/en

[8] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "ArduSim: Accurate and real-time multicopter simulation," *Simul. Model. Pract. Theory*, vol. 87, pp. 170–190, Sep. 2018, doi: 10.1016/j.simpat.2018.06.009.

[9] E. W. Justh and P. S. Krishnaprasad, "A simple control law for UAV formation flying," Maryland Univ. Inst. Syst. Res., College Park, MD, USA, Tech. Rep. TR 2002-38, 2002.

[10] R. L. Lidowski, B. E. Mullins, and R. O. Baldwin, "A novel communications protocol using geographic routing for swarming UAVs performing a search mission," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun.*, Mar. 2009, pp. 1–7.

[11] B. J. O. de Souza and M. Endler, "Coordinating movement within swarms of UAVs through mobile networks," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2015, pp. 154–159.

[12] A. Sivakumar and C. K.-Y. Tan, "UAV swarm coordination using cooperative control for establishing a wireless communications backbone," in *Proc. 9th Int. Conf. Auton. Agents Multiagent Syst. (AAMAS)*, vol. 3. Richland, SC, USA: International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 1157–1164. [Online]. Available: http://dl.acm.org/citation.cfm?id=1838186.1838188

[13] Z. Zhao and T. Braun, "Topology control and mobility strategy for UAV ad-hoc networks: A survey," in *Proc. Joint ERCIM eMobility MobiSense Workshop*, 2012, pp. 27–32.

[14] . Bekmezci, O. K. Sahingoz, and . Temel, "Flying ad-hoc networks (FANETs): A survey," *Ad Hoc Netw.*, vol. 11, no. 3, pp. 1254–1270, May 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870512002193

[15] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors," *Auton. Robots*, vol. 35, no. 4, pp. 287–300, Nov. 2013, doi: 10.1007/s10514-013-9349-9.

[16] I. Bayezit and B. Fidan, "Distributed cohesive motion control of flight vehicle formations," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5763–5772, Dec. 2013.

[17] T. Zeng, M. Mozaffari, O. Semiari, W. Saad, M. Bennis, and M. Debbah, "Wireless communications and control for swarms of cellular-connected UAVs," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, Oct. 2018, pp. 719–723.
[18] R. Rahimi, F. Abdollahi, and K. Naqshi, "Time-varying formation control of a collaborative heterogeneous multi agent system," *Robot. Auton. Syst.*, vol. 62, no. 12, pp. 1799–1805, Dec. 2014.
[19] X. Dong, Y. Zhou, Z. Ren, and Y. Zhong, "Time-varying formation control for unmanned aerial vehicles with switching interaction topologies," *Control Eng. Pract.*, vol. 46, pp. 26–36, Jan. 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066115300289
[20] *ArduSim. Accurate and Real-Time Multi-UAV Simulation*. Accessed: Jan. 30, 2019. [Online]. Available: https://bitbucket.org/frafabco/ardusim/src/master/
[21] L. Meier and QGroundControl. *MAVLink Micro Air Vehicle Communication Protocol*. Accessed: Jan. 30, 2019. [Online]. Available: http://qgroundcontrol.org/mavlink/start
[22] *OMNeT++ Discrete Event Simulator*. Accessed: Jan. 30, 2019. [Online]. Available: https://omnetpp.org/
[23] *NS-2 The Network Simulator*. Accessed: Jan. 30, 2019. [Online]. Available: http://nsnam.sourceforge.net/wiki/index.php/Main_Page,
[24] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore, "Google earth engine: Planetary-scale geospatial analysis for everyone," *Remote Sens. Environ.*, vol. 202, pp. 18–27, Dec. 2017.
[25] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "A methodology for measuring UAV-to-UAV communications performance," in *Proc. 14th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2017, pp. 280–286.

FRANCISCO FABRA received the master's degree in computer and network engineering from the Universitat Politècnica de València (UPV), Spain, where he is currently pursuing the Ph.D. degree with the Networking Research Group (GRC). His research is focused on the simulation and automatic control of UAV swarms.

WILLIAN ZAMORA received the Ph.D. degree in informatics from the Department of Computer Engineering, Univiersitat Politècnica de València (UPV), Spain, in 2018. He is an Associate Professor with the Faculty of Computer Science, Universidad Laica Eloy Alfaro de Manabí, Ecuador. His research interests include crowdsensing, the IoT, UAVs, data science, and machine learning.

PABLO REYES received the B.Sc. degree in computer science and the M.Sc. degree in computer and network engineering from the Universitat Politècnica de València (UPV), Spain, in 2017 and 2018, respectively. His research interests include UAV manufacturing and UAV-based systems, including UAV swarm protocols.

JULIO A. SANGÜESA received the B.Sc. degree in computer science from the University of Zaragoza, in 2008, the M.Sc. degree from the Open University of Catalonia, in 2010, and the Ph.D. degree from the University of Zaragoza, in 2014. He is an Assistant Professor with the University Center of Defense, Zaragoza. His research interests include electric vehicles, V2V and V2I communications, intelligent transportation systems, and smart cities.

CARLOS T. CALAFATE received the degree (Hons.) from the University of Oporto, Portugal, in 2001, and the Ph.D. degree in informatics from the Universitat Politècnica de València (UPV), Spain, in 2006. He is a Full Professor with the Department of Computer Engineering, UPV. His research interests include ad-hoc and vehicular networks, UAVs, smart cities, and the IoT.

JUAN-CARLOS CANO (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in computer science from the Universitat Politècnica de València (UPV), Spain, in 1994 and 2002, respectively. He is a Full Professor with the Department of Computer Engineering, UPV. His current research interests include wireless communications, vehicular networks, mobile ad hoc networks, and pervasive computing.

PIETRO MANZONI (Senior Member, IEEE) received the Ph.D. degree from the Politecnico di Milano, Italy. He is a Full Professor of computer networks with the Universitat Politècnica de València (UPV), Spain. His research activity is focused on networking and mobile systems for the Internet of Things, especially in the contexts of community networks and intelligent transport systems.