

RESEARCH

Open Access



# Enabling resilient UAV swarms through multi-hop wireless communications

David Clerigues<sup>1</sup>, Jamie Wubben<sup>1\*†</sup> , Carlos T. Calafate<sup>1†</sup>, Juan-Carlos Cano<sup>1†</sup> and Pietro Manzoni<sup>1†</sup>

<sup>†</sup>Jamie Wubben, Carlos T. Calafate, Juan-Carlos Cano, and Pietro Manzoni have contributed equally to this work.

\*Correspondence:  
jwubben@disca.upv.es

<sup>1</sup> Computer Engineering,  
Universitat Politècnica de  
València, Camino de Vera, S/N,  
Valencia 46022, State, Spain

## Abstract

In the last decade, the popularity of UAVs, colloquially known as drones, has increased tremendously. Nowadays, drones are used in a wide range of use-cases, such as precision agriculture, surveillance, and photography. Many of these use-cases can be made more efficient if multiple UAVs are used cooperatively (i.e., in a swarm). To achieve this, communication between the UAVs is paramount. To ensure communication, many works rely on the existing infrastructure (e.g., 4G). However, in many rural areas, this infrastructure does not exist. In those cases, an ad hoc (Wi-Fi) network is the most adequate alternative. Yet, due to the limited communication range of Wi-Fi, it is not possible to let UAVs in a swarm to communicate over a long distance. To solve this issue a relay approach is necessary. Despite general solutions to relay messages between (mobile) nodes already exist, many UAV swarms rely on master–slave communication. Thus, a specific solution for this type of communication might be more efficient. Hence, in this work, we propose a strategy to efficiently relay messages for UAV swarms adopting the master–slave communication paradigm. Our approach seeks to introduce a very small message overhead to avoid congestion of the network, and to provide more bandwidth for the actual applications of the UAV swarm. We tested our approach using a realistic UAV simulator called ArduSim. Our results show that our approach is capable of detecting all the nodes in the network within a few seconds. Furthermore, we applied our message relay approach on an existing swarm application (where a swarm of UAVs had to follow a mission), and our results show that, now, the communication range of the UAVs can be much larger, without impacting other aspects of the mission (such as flight time).

**Keywords:** Routing, UAV swarms, ArduSim, Multi-hop, Gateway-based solution

## 1 Introduction

Unmanned aerial vehicles (UAVs) have emerged as autonomous flying robots that can perform different functions remotely. Although in the early stages they had military purposes [1, 2], over the years their field of applicability has broadened significantly, and so, nowadays, drones are capable of performing different jobs, ranging from healthcare delivery [3] to precision agriculture [4], among many others [5].

Although single UAVs are able to perform a plethora of tasks, they suffer from various problems that prohibit them from executing complex tasks efficiently. One of the major issues is the battery lifetime, which limits most of the UAVs to a flight

time of only 20 min. For complex tasks, such as patrolling or recognizing an area affected by a disaster, this is often not sufficient. The obvious solution is deploying more UAVs at the same time, and, in order to obtain the highest efficiency, we want those UAVs to communicate with each other so as to operate in a cooperative fashion. Such a group of coordinated UAVs is commonly referred to as a swarm. Although such a swarm is able to perform more complex tasks (e.g., rescuing people after catastrophes [6, 7], or pesticide spraying [8]), they face several problems in order to operate, including energy consumption, communications, the architecture used, etc. [9].

Communication is a key issue for deploying UAV swarms. In order to let the UAVs work cooperatively, we can rely either on an existing infrastructure (e.g., 4G), or create a local (ad hoc) network. Since in rural areas the coverage of existing infrastructures is low, creating an ad hoc network is often the only option. Wi-Fi (802.11) is often used, since it provides a good trade-off between communication range, power consumption, latency, and practicality. Earlier research [10] has shown that, in open areas, UAVs can communicate up to several hundred meters using Wi-Fi. Although useful, this is not enough for some applications (e.g., in agriculture) where the swarm needs to cover a large area. Hence, we must rely on a relay strategy to increase the communication range.

Relay strategies do of course already exist. However, many UAV swarms rely on master slave communication. This occurs because (i) the decision-making process is easier with one centralized controlling node (i.e., the master), and (ii) it is easier to synchronize the UAVs when decisions are coming from one UAV. This particular information flow does not have the same requirements as a generic approach. Hence, the rerouting algorithm can be made simpler and more efficient. For that reason, we propose a new strategy, tailor-made for master–slave communication. The proposed strategy involves two states: (i) neighbor discovery and (ii) multi-hop network setup. In the first state, each node discovers its neighbors. Each slave then establishes the number of hops and gateways to reach the master node in the second state. Also, we have accumulated messages from slaves during the flight (piggybacking). Through this approach, messages are routed simply, easily, quickly, and efficiently.

For performance assessment, we have used a realistic multi-UAV simulator called ArduSim [11]. We use it to implement our multi-hop swarm protocol, and to test our approach. The results show that it is possible to fly the swarm in different formations and sizes. The lower the number of hops, the better the performance as we achieve: (i) lower congestion, (ii) lower delay, (iii) lower message sending/receiving, and (iv) lower flight time.

The remainder of this paper is structured as follows: Sect. 2 introduces works related to swarm coordination, routing algorithms, and multi-hop approaches. Section 3 details our proposal to achieve an algorithm capable of managing a swarm in multi-hop environments. Afterward, in Sect. 4, we present the simulation environment which we use to validate our proposal. Next, Sect. 5 defines the mission parameters and measures the overhead of our protocol. Moreover, it presents the performance obtained on the swarm using the proposed protocol. Finally, Sect. 6 summarizes the main findings of this work, and refers to future work.

## 2 Related work

In this section, we study different related works focused on (i) routing algorithms, (ii) multi-hop approaches, and (iii) coordination models for swarms or related technologies. We have extensively analyzed various works on routing algorithms. This has allowed us to develop a solution that is both effective and efficient. In addition, the performance of wireless networks has been investigated. Mainly, these investigations have allowed us to evaluate whether or not it is feasible to use this technology in multi-hop environments. Finally, several coordination patterns have been reviewed. These have allowed us to broaden our view on coordination.

### 2.1 Routing algorithms & multi-hop approaches

In deployments where the distances involved are high, radio communications may fail to reach all nodes, and so multi-hopping must be supported via appropriate routing mechanisms. To develop our routing solution, we have extensively analyzed different researches. We have divided the algorithms into two sections: (i) routing algorithms and (iii) algorithms for multi-hop environments.

#### 2.1.1 Routing algorithms

In 1999, Bose et al. [12] explained the initial distributed algorithms for routing which eliminate the need for duplicating packets or memory at individual nodes, while also ensuring that the packet is delivered to its intended destination. These algorithms can be further developed to create broadcasting and geocasting algorithms that also eliminate the need for packet duplication. Furthermore, on-demand multicast routing protocol (ODMRP) was presented by Lee et al. [13]. The protocol utilizes a mesh topology to enable multicast transmissions by integrating the concept of forwarding groups, and employs on-demand techniques to establish routes and manage membership within multicast groups. Malik et al. [14] and Singh et al. [15] conducted a comparison of various routing protocols in vehicular ad hoc network (VANETs), and determined that the selection of routing protocol has a significant impact on packet delivery ratio, end-to-end delay, and transmission overhead. Similarly, Coutinho et al. [16] also evaluated the performance of different routing protocols and concluded that optimized link state routing (OLSR) outperformed on-demand distance vector (AODV) and destination-sequenced distance vector (DSDV) in VANET scenarios. Moreover, Bundel et al. [17] introduced two algorithms that are designed to minimize delays by establishing collision-free links for data aggregation. Compared to existing methods, these algorithms significantly reduce latency. Frey et al. [18] outlined the difficulties involved in developing routing protocols for sensor networks, and emphasized on the essential methods for achieving efficient energy usage and ensuring a successful delivery. Radi et al. [19] discussed the potential benefits of implementing multipath routing in order to enhance network performance by optimizing the utilization of available network resources. Lastly, Amorosi et al. [20] recently examined the optimization of routing problems involving drones, exploring various approaches based on the assumption made about the path taken by a mothership.

#### 2.1.2 Multi-hop routing algorithms

An enhanced multi-hop clustering algorithm (EMCA) was proposed by Qian et al. [21] in 2006. It utilizes multi-hop links for both intra-cluster and inter-cluster communication,

and the scalability of the network significantly improves as its scale grows larger. In addition, Yamao et al. [22] presented a method for dynamic multi-hop transmission that can improve the probability of successful packet delivery by reducing unnecessary transmissions. Moreover, Rehman and Wolf [23] argued that relaying can be a technically suitable option for multi-hop connectivity in wireless ad hoc networks, and compares the performance of a multi-hop version of the IEEE 802.11 MAC protocol with the AODV routing protocol. Next, Kumar et al. [24] proposed a multi-hop clustering algorithm that was randomized and distributed, which successfully addressed the issue of hot spots in heterogeneous Wireless Sensor Networks (WSNs). This algorithm led to a significant increase in network lifetime and more balanced load distribution when compared to existing algorithms. By examining a proposed multi-hop routing protocol, Sharma et al. [25] evaluated how modifying the packet size affected its performance. He discovered that, as the packet size increased, the throughput reached a maximum at a certain point, and then gradually decreased. Biradar et al. [26] examined various multi-hop routing protocols, and concludes that the multi-hop-LEACH (low-energy adaptive clustering hierarchy) protocol consumes less power and exhibits lower latency compared to the other protocols. In addition, Prabha et al. [27] examined the various categories of wireless multi-hop ad hoc networks, and discuss the primary advancements that have been made in this field, which includes opportunistic networks. Furthermore, Pant et al. [28] put forward a multi-hop routing protocol that utilized grid clustering, resulting in a longer lifespan for the network when compared to existing algorithms. Finally, Pinto et al. [29] characterized multi-hop aerial networks of commercial-off-the-shelf UAVs, and derived the optimal number of hops that maximize the end-to-end throughput.

The related works are very specific. Sensor networks have major disadvantages such as (a) limited scalability, (b) high packet loss, (c) high latency, and (d) limited processing capabilities. In addition, VANETs also have disadvantages such as (a) transmission limitations, (b) frequent topology changes, and (c) low infrastructure availability. The analyzed works use different types of algorithms, whether distributed, graph-based, demand-based, etc. All of them attempt to solve most of the mentioned problems under very specific conditions. Our idea, developed based on hops and gateways, allows for a more general approach, and has a number of advantages: (i) reduction of network congestion, (ii) improved routing efficiency, and (iii) greater scalability.

## 2.2 Coordination models

A quite unconventional approach was presented by Parunak et al. [30], which utilizes a computational analog of pheromone dynamics to enable the coordination of multiple UAVs; this method can be implemented autonomously by a UAV. Secondly, Richards et al. [31] discussed solutions for autonomous task allocation and trajectory planning for a fleet of UAVs, and compared two optimization methods that integrate task assignment and path planning to address these challenges. In addition, Souza et al. [32] proposed an algorithm which is designed for coordinating multiple UAVs in a swarm formation, while ensuring efficient utilization of bandwidth in areas covered by mobile networks. Moreover, Yanmaz et al. [33] outlined the difficulties involved in creating cooperative airborne systems made up of drones equipped with onboard sensors, processing, coordination, and communication capabilities, and offered prospective resolutions based on acquired knowledge. Kim et al. [34] suggested a control scheme that enables real-time management of multi-hop

drones even when they are out of the communications range of the ground control system. Chen et al. [35] investigated communication and networking techniques suitable for drone swarms, and proposed an intelligent and robust solution for drone swarms which can effectively improve the robustness to node failures. Lastly, Saffre et al. [36] outlined two approaches to control, direct and indirect, which have applicability in designing graphical user interfaces (GUIs), hence allowing a single operator to coordinate a swarm's movements.

Related work presents three fronts: (a) use of control stations, (b) use of flight operators, and (c) automating swarms. We find that stations bring a number of disadvantages: (i) lack of flexibility, (ii) limitations in autonomy, (iii) infrastructure costs, and (iv) point dependency. In addition, employing a flight operator also has its disadvantages: (i) limitations in control, (ii) risks of human error, and (iii) lack of autonomy. The investigated works present very specific cases, and lack of multi-hop approaches. Hence, our swarm operates automatically and autonomously. In addition, the employed swarm maintains coordination using the master–slave pattern. This mechanism provides: (i) increased accuracy, (ii) increased efficiency, (iii) scalability, and (iv) improved ability to operate in remote areas. With our solution, we provide a coordinated and multi-hop swarm capable of carrying out complex missions without infrastructure support.

### **3 Proposal of a swarm management protocol for multi-hopping**

The main objective of the protocol is to maintain coordination, connectivity, and communication stability in dispersed swarms in a multi-hop environment. Coordination can be achieved with different mechanisms such as negotiation, consensus, and master–slave. We have chosen the master–slave approach for its effectiveness, simplicity, and precise control of actions. In this mechanism, there are two types of nodes: (i) master, which coordinates the swarm, and (ii) slave, who obeys the orders of the master. In order to increase network performance, the master node is always the central node (from a geographical perspective), while the rest are slaves.

Coordinating the swarm in a multi-hop environment is complex. Distancing the nodes may be a requirement to cover a large area, but causes increased packet loss and delays due to poorer communication conditions. In fact, multi-hopping becomes a requirement whenever any of the nodes moves beyond the transmission radius of the master nodes. Hence, intermediate nodes are required to act as relays to carry out such communication, making sure all nodes are reachable. Hence, to ensure communication between the slaves and the master in multi-hop environments, it becomes necessary to differentiate the slaves into two types: (i) intermediate nodes and (ii) leaf nodes. The intermediate nodes are in charge of routing messages. The leaf nodes are the farthest from the master, and so they do not forward incoming messages; yet, in our approach, they are the first slaves to start transmitting.

In order for the proposed protocol coordination mechanism to work, we still need to ensure the stability and connectivity of communications. To do so, we must guarantee: (i) communication between master and all slaves, (ii) routing through intermediate nodes, (iii) maintaining coordination in high-dispersion scenarios, and (iv) avoiding network congestion.

To achieve the aforementioned requirements, the designed protocol has three states: (a) neighbor discovery, to quickly find the nodes within direct radio range, (b) multi-hop network setup, to determine the routes to/from the master, and (c) mission phase, for the actual flight. These are explained in detail in the following subsections.

### 3.1 Neighbor discovery

This is the first state of the protocol. Each node in the swarm sends and receives messages in a loop for a predefined amount of time (see details in Sect. 5). In this way, each node discovers which nodes are within their radio range.

**Algorithm 1** Neighbor discovery state

---

```

startTime ← CurrentTime
while CurrentTime − startTime < establishedTime do
  Send Neighbor Discovery Message (Node ID + State ID)
  msg ← Receive Neighbor Discovery Message (Mandatory State ID)
  if msg ≠ null then
    Neighbors ← msg[senderID]
  end if
end while

```

---

Algorithm 1 shows the behavior during this state. As long as the execution time is less than the established time, a message containing the state identifier and the node identifier is sent; after sending, the channel buffer is read. If there is a neighbor discovery state message in the buffer, it is processed. The processing is simple: first the sender's identifier is extracted, and then this identifier is stored in the neighbor list. Discovering neighbors is crucial for the next state.

### 3.2 Multi-hop network setup

In this phase, each UAV must: (i) know the number of hops to the master, and (ii) know its gateway. In addition, each node has to obey the following rules:

- The master broadcasts a few messages that include the list of all its neighbors.
- Each slave, upon receiving a message, propagates it only if three conditions are simultaneously met:
  1. Its number of hops is higher than that of the message received (to minimize broadcast storm effects).
  2. Its identifier is in the list received (to guarantee that only stable links are used).
  3. It is able to reach nodes not yet included in the list (for the forwarding to be useful); this is achieved by adding its neighbors to the list and increasing the number of hops before retransmitting.

The master has a simple function. It only needs to send a predefined number of messages with (i) its UAV identifier, (ii) the multi-hop network setup phase identifier, and (iii) its neighbor list. Unlike the previous phase, sending a pair of messages still provides improved resilience to loss, and is better than using a timer. This change reduces network congestion and allows the slaves to learn and adjust themselves in a shorter time. Algorithm 2 shows the behavior of the master; for our experiments, the value of  $N_{replicas}$  has been set to four.

**Algorithm 2** Multi-hop network setup—master

---

```

NeighborList  $\leftarrow$  Neighbors
msg = empty JSON
msg.push("NodeID", NodeID)
msg.push("StateID", StateID)
msg.push("NeighborList", NeighborList.all())
for int i = 0; i  $\leq$   $N_{replicas}$ ; i++ do
    Send(msg)
end for

```

---

The slaves execute algorithm 3 in a loop until (i) at least one message has been received, (ii) the last message was received more than  $T_{timeout}$  seconds ago; for our current implementation this value has been set to 500 ms. During this time, the slave listens on the channel. If it receives a message, it ensures that the three conditions are met. If not, it starts again. Once the message has been received, the number of hops is first checked. If its hop count is higher than the number of hops in the message, it moves on to the next rule. Next, it checks if the associated list contains its identifier. Finally, if the node reaches more nodes than those present in the list, it forwards the message. Otherwise, it is a leaf node and exits the loop.

**Algorithm 3** Multi-hop network setup—slave

---

```

lastMessage  $\leftarrow$  Current Time
firstMessageRecv  $\leftarrow$  false
while firstMessageRecv and (Current Time - lastMessage  $\geq$   $T_{timeout}$ ) do
    msg  $\leftarrow$  Receive Multi-hop Setup Message (Mandatory State ID)
    if msg  $\neq$  null then
        recvNodeID  $\leftarrow$  msg[senderID]
        recvNodeHops  $\leftarrow$  msg[hops]
        recvNeighborList  $\leftarrow$  msg[senderID]
        if (recvNodeHops < nodeHops - 1) or (nodeHops == null) then
            nodeHops  $\leftarrow$  ++recvNodeHops
            gateMap  $\leftarrow$  (recvNodeID, recvNodeHops)
        else
            continue ▷ Next while iteration
        end if
        if recvNeighborList not contains nodeID then
            continue ▷ Next while iteration
        end if
        if recvNeighborList contains all neighborList then
            imLeaf  $\leftarrow$  true
            break ▷ Exit while loop
        else
            Send Multi-hop Setup Message (Node ID + State ID + Neighbor List)
            firstMessageRecv  $\leftarrow$  true
            lastMessage  $\leftarrow$  Current Time
        end if
    end if
end while

```

---

Slaves do not have any hop count assigned at the beginning. When they receive the first message, they determine such value by increasing the hop count registered in the incoming message. Notice that the probability of receiving a first message from a node with more hops is negligible, but it can happen. This does not affect the number of hops, but it does affect the sending of messages in the mission. To avoid this, the gates and their hops are checked. If there is a node stored as a gateway, and it has a higher hop count than the current node, it is deleted. Coordination, connectivity, and stability of communications are guaranteed at this point. Finally, it is necessary to reduce channel congestion. The following sections explain the mission, and how to reduce network congestion.

### 3.3 Reducing and optimizing message delivery/reception

Piggybacking is the data transmission technique used at the link layer to improve performance. Its goal is to group an acknowledgment together with the next packet sent. This mechanism offers a decrease in packet transmissions, thereby improving efficiency. Notice that, when using multi-hop forwarding, the number of messages sent, and the channel congestion, may tend to increase. In addition, we do not ensure that the messages are received by the destination. This scenario is applicable when a message has a destination. The intermediate node will piggyback the message if, and only if, (i) the message it has to send has the same destination as the received message, and (ii) both messages belong to the same state. If these conditions are not met, it sends the received message without piggybacking its own message. Another kind of message is one that has no destination (broadcast), which is typically sent by the master, being addressed to all the slaves. In this case, piggybacking is not applicable. By implementing the piggybacking behavior on our protocol, we have managed to reduce the number of messages sent and the channel congestion, while increasing the efficiency during the course of the mission.

### 3.4 Mission

This mission consists of reaching a series of waypoints. To reach these waypoints, nodes alternate between two sub-states: (i) *Waypoint Reached* and (ii) *Move to waypoint*; such states are repeated in a loop until the last waypoint is reached.

In the *Waypoint Reached* sub-state, the master waits to receive at least one message from each slave in the swarm. This message contains: (i) the message identifier, (ii) a phase state identifier, and (iii) an acknowledgement of arrival. When this condition is fulfilled, it switches to the *Move To Waypoint* state. On the other hand, the slaves differ in their behavior. For instance, a leaf node starts by transmitting *Waypoint Reached* messages. In addition, it only receives *Move To Waypoint* messages. Upon receiving such a message, it changes its status to *Move To Waypoint*. Intermediate slaves do not transmit messages until they receive a message from the leaf or its neighbor. At this point, they can send their own *Waypoint Reached* message. When a message is received, it can be associated to either the *Waypoint Reached* or the *Move To Waypoint* sub-state. For *Waypoint Reached* cases, the node retransmits this message. On the other hand, if it is a *Move To Waypoint* message, it also retransmits such message before transitioning to the *Move To Waypoint* state.



The *Move To Waypoint* state is simpler. The master sends messages to the slaves to tell them to move forward. The message structure consists of: (i) the message identifier, (ii) a phase identifier, and (iii) the command to move. The slave nodes read in the channel buffer. If they receive a *Move To Waypoint* message, they retransmit it. This last step does not apply to leaves. The main objective of this mission is to check the correct coordination of the swarm, and to measure: (i) the channel congestion and (ii) the mission duration.

## 4 Methods

UAV development tends to rely on two alternative strategies: real deployment or simulation experiments. Researchers tend to simulate, since using physical devices can involve significant monetary and material resources. On the other hand, simulations allow achieving a more sophisticated and optimal product, and possibly bring it to a real environment in a safer way.

There is a wide range of simulators on the market today. In this work, we relied on the ArduSim simulator [11]. This simulation tool supports real-time simulations, being an innovative feature in the market. Communications between devices are established through the use of a virtual wireless channel that emulates communications in shared environments in real time. The simulator, being open source, allows different users to create their own solutions, from protocols to specific communication channels. Below, we explain the most relevant points related to this project. For more details, we recommend reading [11].

### 4.1 Physical layer model & communication channels

As stated before, ArduSim is a multi-UAV simulator. In order to bring our simulator close to reality, we took great care of simulating the communications between the UAVs. The most straightforward way to provide communication between real UAVs is to rely on broadcasting (in an ad hoc network) using UDP. However, UDP packets can get lost. We can model this behavior to various degrees of accuracy. In general, the higher the accuracy, the longer it will take to calculate whether a packet will get lost. Since the communication between the UAVs is a central part of our simulator, we provide three different models:

- *Unrestricted*. It uses an ideal medium where data packets always arrive to all possible destinations (basic model).
- *Fixed range*. Data packets arrive to another UAV only if the distance between them is lower than the defined threshold (simple model).
- *Realistic 802.11a with 5dBi antenna*. The probability that a data packet is received by another UAV depends on the distance between the UAVs. This realistic model is obtained from real experiments where the packet loss rate between two UAVs was measured using a Wi-Fi ad hoc network link in the 5 GHz band. Out of these experiments, we derived a model for the package loss as a function of the distance ( $x$ ) between the UAVs. Beyond 1350 ms, we consider that packet losses reach 100%, and for smaller distances the package loss is modeled by  $y = 5.335 \cdot 10^{-7} \cdot x^2 + 3.395 \cdot 10^{-5} \cdot x$ . Furthermore, in this model, we include carrier sensing functionality.

In all the three models explained above, we are also implementing carrier sensing (as a real network card would do) to ensure that one UAV will not start its broadcast while another UAV is already sending its information. Furthermore, we also consider packet collisions, the transmission time, and the size of the input buffers of the network card.

#### 4.2 Available flight formations

Finally, within the simulator, the user has a defined set of flight formations to structure the swarm, and thus carry out the established mission. The following list shows the different formation types available:

- **Linear formation:** it positions each UAV on a straight line. Each identifier is assigned from left to right, and the master node is placed in the center.
- **Random formation:** as its name suggests, it distributes the drones randomly around the master. This formation has been useful for performing certain checks with the previously mentioned protocols. Due to its inherent nature, this formation hinders the development of a coherent multi-hop solution.
- **Matrix formation:** it places the master node at the center of the swarm and, consequently, the slave nodes are placed around it to form a square matrix that scales as the size of the swarm increases.
- **Circular formation:** it distributes the slaves around the master by drawing a circumference following a clockwise direction. The limitations of this formation are given by the communications range; therefore, it is not optimal for applying the multi-hop approach.

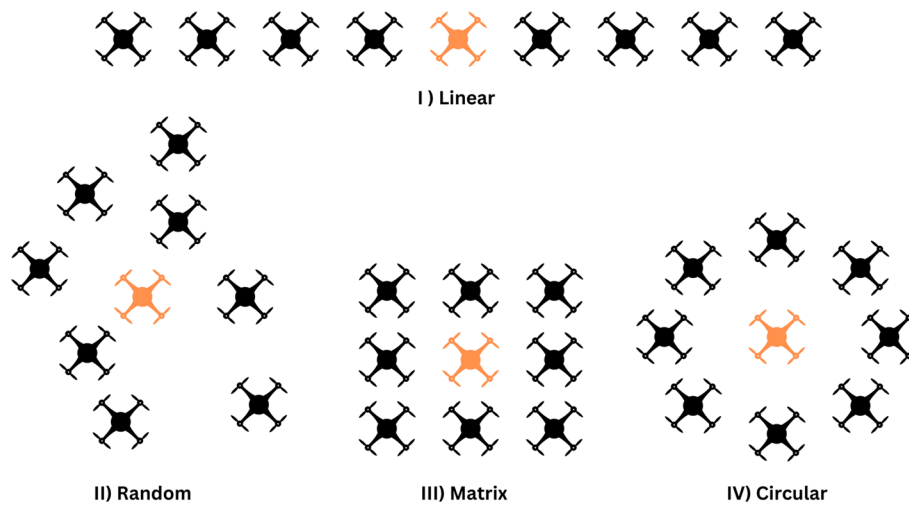
### 5 Results and discussion

The simulator has a wide variety of configurable parameters. For this article, we have used collision detection and carrier sensing. The formations used are linear and matrix. In each we have performed tests for 2, 4, and 6 hops. Messages are transmitted every 500ms (two messages per second) and received every 25ms to avoid saturating the network. The separation between nodes is 240 m. The communication range has been limited to 300 m. This limitation forces the swarm to have each node with the minimum number of neighbors (worst-case scenario). Stressing the network to the maximum allows us to check if the proposed solution is able to overcome the worst-case conditions. The following list specifies the parameters:

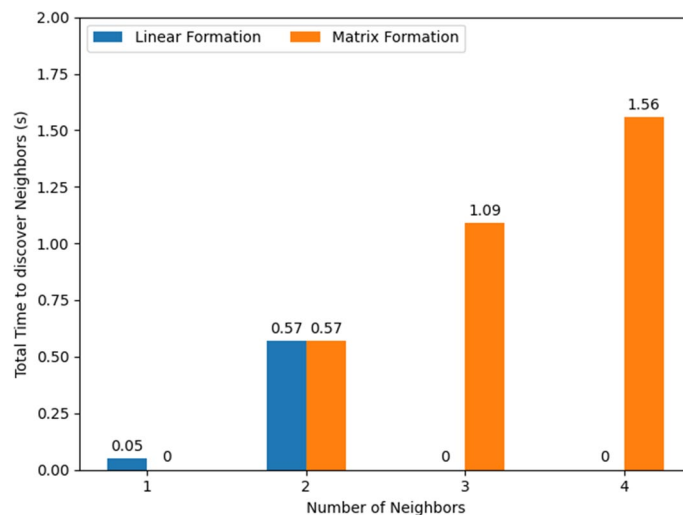
- **Formations:** linear and matrix
- **Communication range:** 300 ms
- **Distance between nodes:** 240 ms
- **Maximum distance between leafs and master:** 2, 4 and 6 hops
- **Number of drones (linear/matrix):** 5/9, 9/25, 13/49
- **Sending time:** 500 milliseconds
- **Receiving time:** 25 milliseconds

### 5.1 Pre-mission phase time overheads

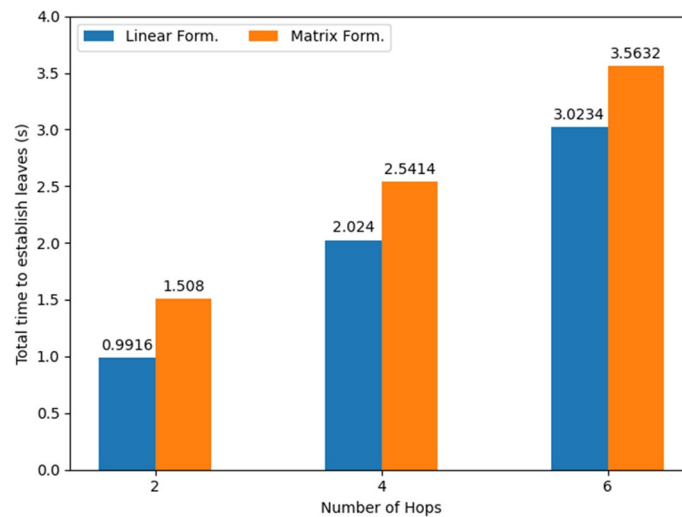
Before executing the mission, the neighbor discovery and multi-hop network setup states are executed. Neighbor discovery requires some time to define the neighbors in each node (Fig. 1). We have run the algorithm for 10 s. Here, we have captured precisely what each node requires to discover its neighbors. In Fig. 2, we observe the times in both formations. In linear formation, the leaves have one neighbor, while the master and the rest of the slaves have 2. In the matrix formation, the leaves have two neighbors, their neighbors have 3, and the remaining nodes have 4. The minimum time to complete this state is 50 ms, and the maximum is 1.5 s. We have fixed the timeout at 2 s to guarantee that all nodes are able to discover their neighbors.



**Fig. 1** Flight Formations. Different flight formations used for experiments



**Fig. 2** Multi-hop network setup overhead. The network setup overhead for linear and matrix formation depending on the number of neighbors



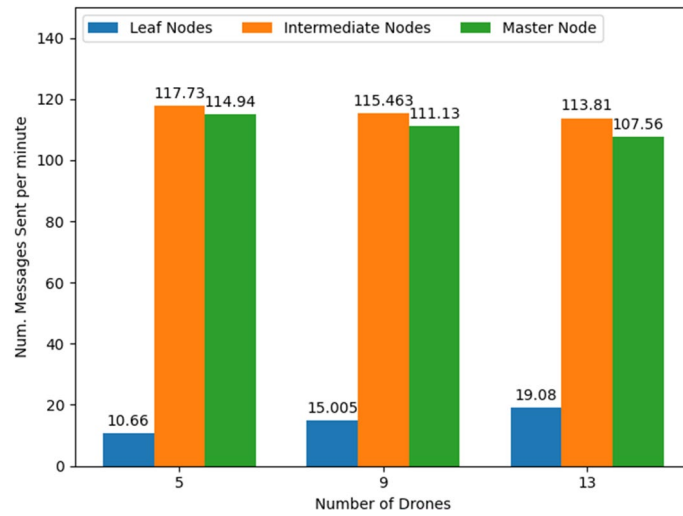
**Fig. 3** Multi-hop network setup overhead. The network setup overhead for linear and matrix formation depending on the number of hops

We have measured how long it takes to establish the leaves to measure the time overhead on the multi-hop network setup. Messages are sent every 500 ms. Therefore, two hops are theoretically reachable within 1 s. Figure 3 shows the expected behavior. Arrayed leaves take longer as they require more neighbors in the leaves. The minimum time to complete this state is 900 ms, and the maximum is 3.5 s. We have 950 ms as the minimum, and 5 s as the maximum time to establish the multi-hop network in the swarm.

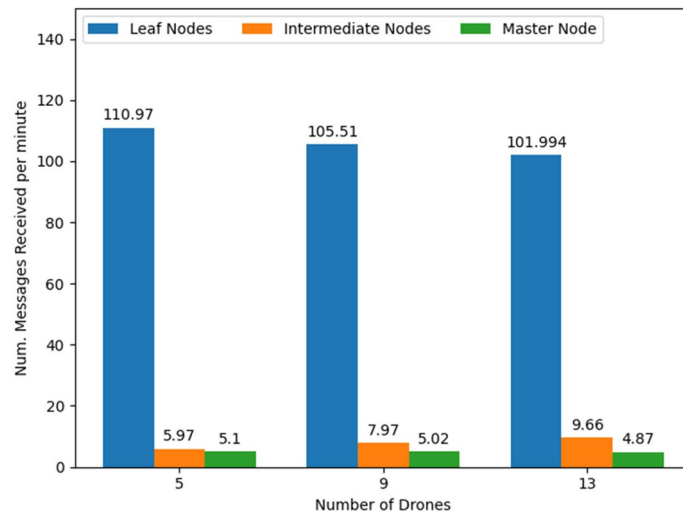
## 5.2 Linear formation performance

In Fig. 4, different behaviors are observed: (i) the leaves send more messages when the number of drones grows, (ii) the rest of the slaves decrease their message sending as the swarm grows, and (iii) the master also decreases its message sending. In the *Waypoint Reached* state, the leaf must communicate with the master, and its message must arrive in order to enable piggybacking messages from the intermediate nodes. As the number of hops increases, packet loss also increases, so it must send more packets to increase resilience to loss. The intermediate nodes experience a decrease in the number of messages received. Hence, they will only forward the few that reach them. Likewise, the master will receive even fewer packets. This will result in less forwarding events. On the other hand, the *Move To Waypoint* state only involves the intermediate nodes and the master. This is the reason for the large difference detected in the sending of messages.

The reception of messages has an inverse behavior, as shown in Fig. 5. The leaf nodes are the ones that listen all the time. As we increase the number of hops, the reception decreases, meaning that the master node receives fewer messages. In addition, with more hops, the intermediate nodes become more saturated; in particular, intermediate nodes in the *Waypoint Reached* state experience more congestion than the rest; in fact, since they are responsible for reading messages from both states, more channel congestion and resource consumption takes place.



**Fig. 4** Sent messages per minute for the linear formation. Number of messages send per minute in the linear formation showing the difference between leaf nodes, intermediate nodes, and the master node

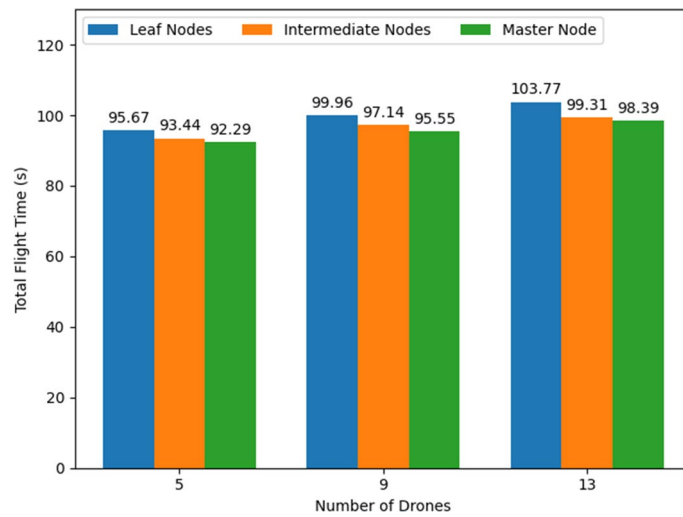


**Fig. 5** Received messages per minute for the linear formation. Number of messages received per minute in the linear formation showing the difference between leaf nodes, intermediate nodes, and the master node

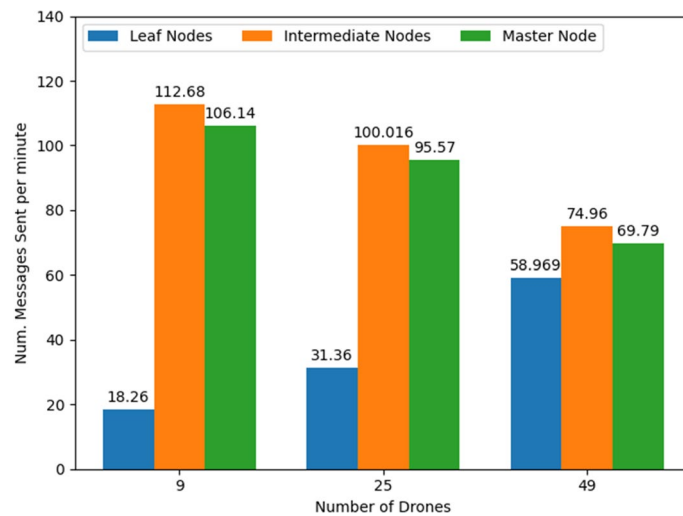
Finally, Fig. 6 shows the total flight time. In all cases, the same behavior is observed. The master node finishes first, the intermediate nodes follow, and the leaves are the last ones to finish as expected. Also note the time difference. Theoretically, two hops are one second. Therefore, a theoretical difference of 1, 2, and 3 s is expected in each trial. However, as observed in our experiments, this time increases.

### 5.3 Matrix formation performance

Figure 7 shows that the matrix formation behaves similarly to the linear formation regarding the sending of messages. However, message loss levels are higher. In the latter case, it is observed how the amount of messages sent by the leaves almost equals the



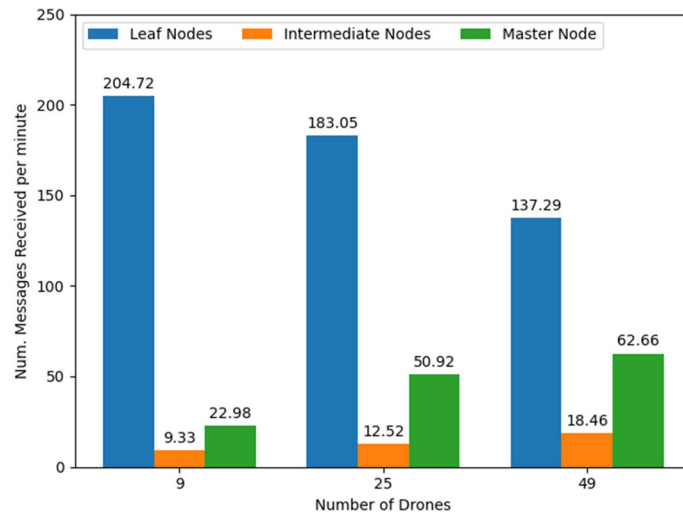
**Fig. 6** Linear formation total flight time. The total flight time in the linear formation, showing the difference between leaf nodes, intermediate nodes, and the master node



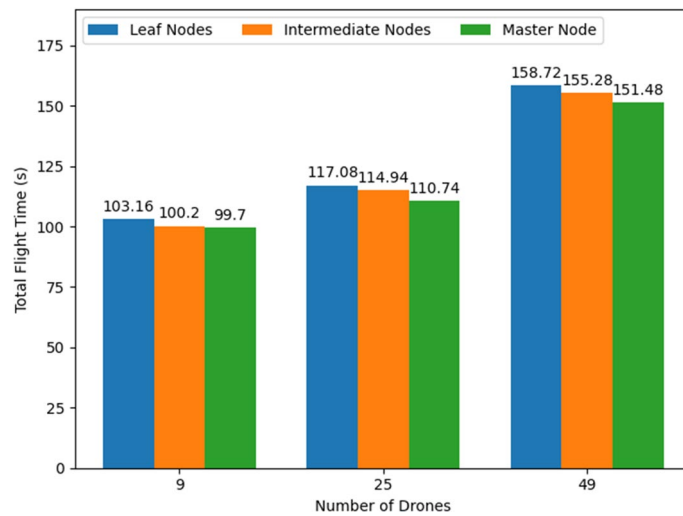
**Fig. 7** Sent messages per minute for the matrix formation. Number of messages send per minute in the matrix formation showing the difference between leaf nodes, intermediate nodes, and the master node

others. In the matrix formation, there are more routes available to forward messages. Therefore, it is expectable to experience a higher level of loss.

Message reception increases in this formation. Figure 8 shows the same behavior in the leaf nodes. However, the master node, instead of experiencing a reduction of received messages, experiences an increase. This behavior is caused by packet loss. The master node, during the *Waypoint Reached* state, must register all 48 slave identifiers. An increase in reception indicates that packet loss causes certain identifiers to take longer to arrive. This indicates that the intermediate nodes suffer more congestion in this formation due to the increased number of neighbors, which are prone to cause collisions on the channel.



**Fig. 8** Received messages per minute for the matrix formation. Number of messages received per minute in the matrix formation showing the difference between leaf nodes, intermediate nodes, and the master node



**Fig. 9** Matrix formation total flight time (s). The total flight time in the matrix formation showing the difference between leaf nodes, intermediate nodes, and the master node

The flight time is similar to the Linear formation. In the first case, a 1-s leaf-master lag is expected. However, tests show that, in reality, this time lag is higher. In the second case, the expected time lag is 2 s; yet, the result obtained is three times greater than expected. Finally, the scenario with 49 drones should have a time lag of 3 s. However, the result obtained is almost three times the theoretical time lag. These results, obtained from Fig. 9, are associated to the high packet loss.

### 6 Conclusions and future work

Drone swarms have great potential. Yet, swarm missions must maintain cohesion and coordination to be successful. Achieving such goals can become more complex in multi-hop environment due to channel losses and associated delays. To address this challenge,

in this paper, we propose a swarm management protocol for flying in multi-hop environments. The protocol consists of three states: (i) neighbor discovery, (ii) multi-hop network setup, and (iii) mission. The first two states add a minimum time overhead of 950 ms and a maximum of 5 s. Tests show how increasing the number of hops causes an increase in packet loss. This causes an increase in the send rate, a decrease in the receive rate, and a temporal increase in the flight time. However, the swarm has been able to successfully fly in both linear (5, 9, 13 drones) and matrix (9, 25, 49 drones) formations.

As future work, we want to further investigate the effectiveness of our solutions by carrying out experiments in a real environment. In particular, we plan to determine the resilience levels achieved under adverse channel conditions, including the presence of obstructions, and highly fluctuating signal-to-noise ratios. Furthermore, we are interested in comparing our proposed solution with other existing related works.

#### Abbreviations

AODV	On-demand distance vector
DSDV	Destination-sequenced distance vector
EMCA	Enhanced multi-hop clustering algorithm
ODMRP	On-demand multicast routing protocol
OLSR	Optimized link state routing
VANETs	Vehicular ad hoc network
WSNs	Wireless sensor networks

#### Acknowledgements

We would like to thank the reviewers for their suggestions for improvement. Thanks to them, we have been able to improve the content and quality of the research.

#### Author Contributions

DC carried out the coding, conducted experiments, analyzed results, and played a key role in drafting the manuscript. JW contributed to the conceptualization of the idea, contributed to code development, and participated in both writing and reviewing the paper. CC conceptualized the idea, reviewed the paper, secured funding, and provided supervision. JC reviewed the manuscript and provided supervision. PM reviewed the paper and offered supervision. All authors actively engaged in the study, and read and approved the final manuscript.

#### Funding

This work is derived from R & D Projects PID2021-122580NB-I00 and SRTC1900C007159XV0, funded by MCIN/AEI/10.13039/501100011033 and “ERDF A way of making Europe.”

#### Data availability

All of our code is open source and a stable version of our code can be found on the following Github page. <https://github.com/GRCDEV/ArduSim>.

#### Declarations

##### Competing interests

The authors declare that they have no competing interests.

Received: 2 January 2024 Accepted: 7 May 2024

Published online: 16 May 2024

#### References

1. J.M. Sullivan, Revolution or evolution? the rise of the uavs. In: *Proceedings. 2005 International symposium on technology and society, 2005. Weapons and wires: prevention and safety in a time of fear*. ISTAS 2005., pp. 94–101 (2005). <https://doi.org/10.1109/ISTAS.2005.1452718>
2. Z. Xiaoning, Analysis of military application of uav swarm technology. In *2020 3rd International Conference on Unmanned Systems (ICUS)*, pp. 1200–1204 (2020). <https://doi.org/10.1109/ICUS50048.2020.9274974>
3. J.E. Scott, C.H. Scott, Drone delivery models for healthcare. In: *Hawaii International Conference on System Sciences* (2017)
4. L. Negash, H.-Y. Kim, H.-L. Choi, Emerging uav applications in agriculture. in *2019 7th International Conference on Robot Intelligence Technology and Applications (RiTA)*, pp. 254–257 (2019). <https://doi.org/10.1109/RITAPP.2019.8932853>
5. L.J. Fennelly, M.A. Perry, Appendix 16a - unmanned aerial vehicle (drone) usage in the 21st century. In: Davies, S.J., Fennelly, L.J. (eds.) *The Professional Protection Officer (Second Edition)*, Second edition edn., pp. 183–189.



- Butterworth-Heinemann, Boston (2020). <https://doi.org/10.1016/B978-0-12-817748-8.00050-X>. <https://www.sciencedirect.com/science/article/pii/B978012817748800050X>
6. M. Erdelj, E. Natalizio, Uav-assisted disaster management: applications and open issues. In: 2016 international conference on computing, networking and communications (ICNC), pp. 1–5 (2016). <https://doi.org/10.1109/CCNC.2016.7440563>
  7. R. Arnold, J. Jablonski, B. Abruzzo, E. Mezzacappa, Heterogeneous uav multi-role swarming behaviors for search and rescue. In: 2020 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA), pp. 122–128 (2020). <https://doi.org/10.1109/CogSIMA49017.2020.9215994>
  8. B.S. Façal, F.G. Costa, G. Pessin, J. Ueyama, H. Freitas, A. Colombo, P.H. Fini, L. Villas, F.S. Osório, P.A. Vargas, T. Braun, The use of unmanned aerial vehicles and wireless sensor networks for spraying pesticides. *J. Syst. Architect.* **60**(4), 393–404 (2014). <https://doi.org/10.1016/j.sysarc.2014.01.004>
  9. D. Rojas Vilorio, E.L. Solano-Charris, A. Muñoz-Villamizar, J.R. Montoya-Torres, Unmanned aerial vehicles/drones in vehicle routing problems: a literature review. *Int. Trans. Oper. Res.* **28**(4), 1626–1657 (2021). <https://doi.org/10.1111/itor.12783>
  10. F. Fabra, C.T. Calafate, J.C. Cano, P. Manzoni, A methodology for measuring uav-to-uav communications performance. in 2017 14th IEEE annual consumer communications & networking conference (CCNC), pp. 280–286 (2017). <https://doi.org/10.1109/CCNC.2017.7983120>
  11. F. Fabra, C.T. Calafate, J.C. Cano, P. Manzoni, ArduSim: accurate and real-time multicopter simulation. *Simul. Model. Pract. Theory* **87**, 170–190 (2018). <https://doi.org/10.1016/j.simpat.2018.06.009>
  12. P. Bose, P. Morin, I. Stojmenovic, J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Netw.* **7**, 609–616 (1999)
  13. S.-J. Lee, W. Su, M. Gerla, On-demand multicast routing protocol in multihop wireless mobile networks. *Mobile Netw. Applicat.* **7**, 441–453 (2002)
  14. S. Malik, P.K. Sahu, A comparative study on routing protocols for vanets. *Heliyon* **5**, e02340 (2019)
  15. S. Singh, P. Kumari, S. Agrawal, Comparative analysis of various routing protocols in vanet. in 2015 Fifth International Conference on Advanced Computing & Communication Technologies, 315–319 (2015)
  16. B.V. Coutinho, E.C.G. Wille, H.I.D. Monego, Performance of routing protocols for vanets: a realistic analysis format. in *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication* (2015)
  17. B. Bundel, V.K. Menaria, Low latency based efficient aggregation scheduling in multihop wireless sensor network. in 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN), 124–128 (2016)
  18. H. Frey, S. Rührup, I. Stojmenovic, Routing in wireless sensor networks. In: *Guide to wireless sensor networks* (2009)
  19. M. Radi, B. Dezfouli, K.A. Bakar, M. Lee, Multipath routing in wireless sensor networks: survey and research challenges. *Sensors (Basel, Switzerland)* **12**, 650–685 (2012)
  20. L. Amorosi, J. Puerto, C. Valverde, Coordinating drones with mothership vehicles: the mothership and drone routing problem with graphs. *Comput. Oper. Res.* **136**, 105445 (2021)
  21. Y. Qian, J. Zhou, L. Qian, K. Chen, Highly scalable multihop clustering algorithm for wireless sensor networks. in 2006 International conference on communications, circuits and systems Vol 3, pp. 1527–1531 (2006)
  22. Y. Yamao, Y. Kadowaki, K. Nagao, Dynamic multi-hopping for efficient and reliable transmission in wireless ad hoc networks. in 2008 14th Asia-Pacific Conference on Communications, 1–4 (2008)
  23. H.-u. Rehman, L.C. Wolf, A multihop IEEE 802.11 mac protocol for wireless ad hoc networks. in 2009 29th IEEE International Conference on Distributed Computing Systems Workshops, 432–439 (2009)
  24. D. Kumar, R.B. Patel, Multi-hop data communication algorithm for clustered wireless sensor networks. *Int. J. Distrib. Sensor Netw.* **7**(1), 984795 (2011)
  25. N. Sharma, Impact of varying packet size on multihop routing protocol in wireless sensor network. (2014)
  26. R.V. Biradar, D.S.P. Sawant, D.R.R. Mudholkar, D.V.C. Patil, Multihop routing in self-organizing wireless sensor networks. (2011)
  27. C. Prabha, S. Kumar, R.K. Khanna, Wireless multi-hop ad-hoc networks: a review. *IOSR J. Comput. Eng.* **16**, 54–62 (2014)
  28. M.K. Pant, B. Dey, S. Nandi, A multihop routing protocol for wireless sensor network based on grid clustering. in 2015 Applications and Innovations in Mobile Computing (AIMoC), pp. 137–140 (2015)
  29. G.-H. Kim, J.-C. Nam, I. Mahmud, Y.-Z. Cho, Multi-drone control and network self-recovery for flying ad hoc networks. in 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 148–150 (2016). <https://doi.org/10.1109/ICUFN.2016.7537004>
  30. H.V.D. Parunak, L.M. Purcell, O'Connell, R.V.: Digital pheromones for autonomous coordination of swarming uav's. (2002)
  31. A.G. Richards, J.S. Bellingham, M. Tillerson, J.P. How, Coordination and control of multiple uavs. (2002)
  32. B.J.O. Souza, M. Endler, Coordinating movement within swarms of uavs through mobile networks. in 2015 IEEE International conference on pervasive computing and communication workshops (PerCom Workshops), 154–159 (2015)
  33. E. Yanmaz, M. Quaritsch, S. Yahyanejad, B. Rinner, H. Hellwagner, C. Bettstetter, Communication and coordination for drone networks. in: *International ICST conference on Ad Hoc networks* (2016)
  34. G.-H. Kim, J.-C. Nam, I. Mahmud, Y.-Z. Cho, Multi-drone control and network self-recovery for flying ad hoc networks. in 2016 Eighth international conference on ubiquitous and future networks (ICUFN), pp. 148–150 (2016). <https://doi.org/10.1109/ICUFN.2016.7537004>
  35. W. Chen, J. Liu, H. Guo, N. Kato, Toward robust and intelligent drone swarm: challenges and future directions. *IEEE Network* **34**, 278–283 (2020)
  36. F. Saffre, H. Hildmann, H. Karvonen, The design challenges of drone swarm control. In: *Interacción* (2021)

## Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.