Drone Systems and Applications

OPEN ACCESS | Research Article

Network analysis of decentralized fault-tolerant UAV swarm coordination in critical missions

Indu Chandran and **Kizheppatt Vipin**

Department of Electrical and Electronics Engineering, BITS Pilani K K Birla Goa Campus, Goa, India

Corresponding author: **Indu Chandran** (email: p20200055@goa.bits-pilani.ac.in)

Abstract

Unmanned aerial vehicles (UAVs) have gained prominence across various sectors for their versatile applications. While their advantages are evident, addressing concerns associated with their deployment is essential to ensure reliability. This study presents an innovative approach for coordinating a group of UAVs in aerial survey missions. The decentralized strategy presented in this article allow UAVs to self-organize into linear formation, optimize their coverage paths, and adapt to agent failures, thereby ensuring efficient and adaptive mission execution. The strategy has been tested and validated on two different platforms: the inter-UAV communication performance is evaluated on NS-3 simulator to measure metrices such as packet delivery ratio, throughput, delay, and routing overhead within the UAV swarms, while mission efficiency and fault tolerance is analyzed on robot operating system framework, and visualized on Gazebo simulator with real-time parameters. Through experimental results, we show that, after proper tuning of control parameters, the approach succeeds in flock formation with high level of fault tolerance, offering higher efficiency in terms of mission time, transmission delay, packet delivery rate, and control overhead, when compared to the benchmark approaches.

Key words: unmanned aerial vehicles, fault tolerance, decentralized, ROS, NS-3, flock formation

1. Introduction

Unmanned aerial vehicles (UAVs), commonly referred to as drones, have experienced remarkable growth and adoption across various sectors in recent years. Their versatility, efficiency, and capability to gather real-time data have made them indispensable tools in industries ranging from agriculture and infrastructure to film production and environmental conservation (Bürkle et al. 2010; Everaerts 2014; Luo et al. 2019; Calafate and Tropea 2020). As businesses and organizations recognize the numerous benefits, the market for UAV technology has experienced exponential growth. This surge can be attributed to technological advancements, including improvements in flight autonomy, payload capacity, sensor capabilities, and data processing, which have expanded the scope of UAV applications and enhanced their performance across industries. Amid their widespread applications, perhaps one of the most significant roles of UAVs emerges in disaster missions. In these critical scenarios, UAVs serve as invaluable assets, providing crucial support to emergency responders, aiding in search and rescue operations, and facilitating rapid assessment of disaster affected areas. Equipped with advanced sensors and imaging technology, UAVs offer unparalleled capabilities for rapid assessment and situational awareness in disaster zones. Their ability to access remote or inaccessible locations allows responders to quickly gather essential data, assess damage, and identify potential hazards, thereby expediting response efforts and minimizing risks to human life. As the technology continues to advance and the regulatory landscape evolves, the importance of UAVs in disaster missions is expected to further increase, solidifying their position as indispensable tools for emergency management and humanitarian efforts.

In the context of disaster response, UAV operations often employ various architectures, and each network architecture presents unique challenges in implementing and coordinating UAVs for area surveillance. The choice of architecture includes centralized, decentralized, and distributed; and depends on the specific requirements of the mission, including scalability, fault tolerance, and autonomy. Centralized architectures involve a single control center managing multiple UAVs, providing centralized coordination and decision-making. This scheme offers simplicity and ease of control but may face challenges such as single-point failures, scalability issues, limited adaptability, and resilience in dynamic environments, thus limiting their effectiveness in disaster response missions. Distributed UAV networks offer increased robustness by distributing decision-making across multiple nodes, reducing vulnerability to single points of failure. While distributed architectures offer increased resilience and flexibility compared to centralized approaches, they also present certain drawbacks that can impact their suitability for disaster response missions. One significant challenge is achieving consensus among distributed nodes regarding task allocation, navigation, and data sharing without relying on a central authority. In dynamic and unpredictable disaster environments, maintaining synchronization and coherence among individual UAVs can be particularly challenging, leading to inefficiencies and suboptimal resource allocation. Variations in environmental conditions, task priorities, or individual UAV capabilities can lead to inconsistencies or conflicts in decision-making and coordination, potentially impacting the effectiveness of surveillance efforts. Additionally, distributed architectures may suffer from increased complexity in managing interactions and dependencies among multiple autonomous nodes, which can result in coordination overhead and reduced efficiency. Also, ensuring robustness to UAV failures or malicious attacks becomes more challenging in distributed architectures, as there is no central control unit to mitigate the impact of individual node failures. Decentralized architectures distribute control among multiple UAVs, allowing for greater flexibility and redundancy. This architecture enables UAVs to operate autonomously or collaboratively, adapting to dynamic environments and improving responsiveness.

Flight formations holds significant importance in UAV surveillance missions. The line formation offers distinct advantages over other formations, particularly in scenarios where the primary objective is to efficiently cover large areas such as disaster zones. One significant advantage of employing a line formation is its inherent simplicity and ease of coordination. By arranging UAVs in a linear configuration, it streamlines the management of flight paths and minimizes the complexity of navigation, thereby reducing the risk of coordination errors. Moreover, the line formation excels in optimizing resource utilization and flight efficiency. With UAVs flying in a straight line, there is minimal deviation from the designated path, resulting in a more direct and expedient coverage of the area. This efficient utilization of resources translates to reduced operational costs and potentially shorter mission durations, which are crucial considerations in disaster response efforts where time is of the essence. In a distributed architecture, when a line formation is established and a UAV within the formation fails, there exists a risk of network partitioning, where the remaining UAVs may become disconnected or unable to communicate effectively with each other. This partitioning can disrupt the coordination and collaboration among UAVs, leading to inefficiencies and potentially compromising the effectiveness of surveillance efforts; whereas in a decentralized architecture, the risk of network partitioning due to UAV failure is mitigated through self-organization and adaptive behavior. Decentralized architectures typically leverage principles of local interactions and collective decision-making, allowing UAVs to dynamically adjust their formations and reconfigure their communication links in response to changes in the network topology. This inherent adaptability makes decentralized architectures well-suited for dynamic and unpredictable environments such as disaster response missions, where the ability to maintain connectivity and coordination in the face of failures or disruptions is essential for mission success.

Despite advancements in decentralization schemes (Huang et al. 2021; Zhou et al. 2023), ensuring robustness in UAV missions against faults and failures is still an area requiring significant attention. Enhancing fault tolerance mechanisms is essential to bolster the effectiveness of UAV missions in disaster scenarios, where quick and reliable response can make a substantial difference in saving lives and mitigating damages. In this study, we consider a disaster monitoring application and introduce a communication approach for the formation and maintenance of a group of UAVs tasked with an aerial survey mission of the disaster site. The UAVs are equipped with monitoring sensors, such as cameras, and their primary objective is to comprehensively cover a specific terrain area, subsequently relaying all collected sensor data to a base station. The proposed solution follows a decentralized approach such that UAVs have the capability of (i) self-organizing in a formation with distinct characteristics; (ii) determining an optimal path for covering the given area, thereby minimizing the mission duration and, (iii) identifying agent failures within the swarm and dynamically planning the path to account for the uncovered area by the faulty agents. This approach ensures the efficient and adaptive execution of aerial survey missions, offering the benefits of decentralized coordination, optimal coverage, and fault tolerance. The effectiveness of our proposed strategy has been tested and validated on two different platforms. In the first set of simulations, we examine swarm efficiency in terms of mission time and fault tolerance, on robot operating system (ROS) framework. The ROS platform facilitates interaction among the agents through ROS publisher/subscriber mechanisms on various ROS topics. This allows for seamless coordination and data exchange between UAVs. Furthermore, the flight dynamics and behavior of the agents are visualized using the Gazebo environment, providing valuable insights into their real-world performance. The second set of simulations aim at assessing the data traffic in the network on Network Simulator-3 (NS-3). This evaluation involves the analysis of crucial performance metrics, including packet delivery ratio, end-to-end delay, and routing overhead. The system model is detailed in the subsequent sections and experimental simulations are presented for different node densities to evaluate the performance of the proposed strategy.

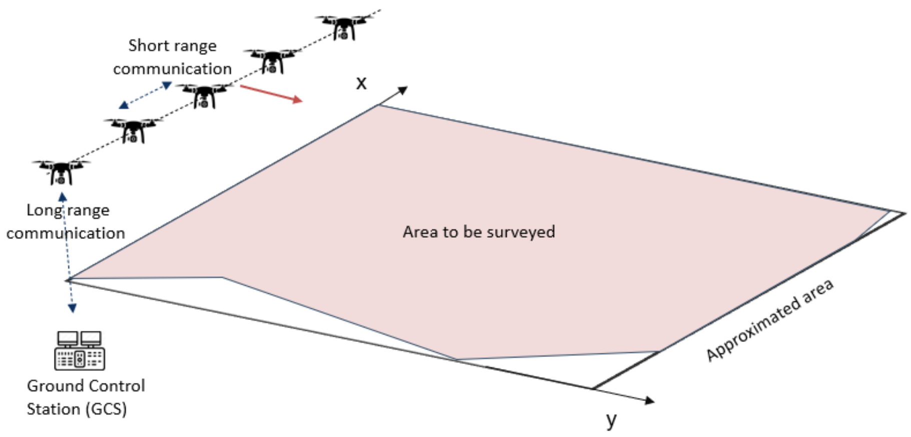
The paper is structured as follows. Section II provides a background on the topic and conducts an analysis of the related literature in the field, offering a comprehensive overview of the existing research and its relevance to the current study. Section III delves into the system model considered in our work, outlining the methodology and framework, and a brief on the simulation environment is presented in section IV. In section V, experimental results are analyzed, emphasizing on the performance and challenges of the proposed strategy, and section VI concludes the paper based on the findings presented in the preceding sections.

2. Related work

Aerial survey and monitoring applications involve a range of challenges that demand specific algorithms and techniques for mission execution (Basir et al. 2021; Wang et al. 2023). These challenges encompass aspects such as formation and maintenance of UAV flocks, establishing communication between UAVs, and developing algorithms for efficient area coverage and path planning. Many of the proposed solutions fall under the domain of multi-agent systems (MASs), viewing a group of UAVs as interacting autonomous agents (Van Dyke Parunak et al. 2003). A dynamic mission planning for UAV swarms is presented in Wei et al. (2013) where the authors introduced a hybrid centralized distributed control framework for mission assignment and scheduling, designed to adapt to changing environmental conditions and missions. With similar objectives, our work, however, considers a decentralized behavior of agents coordinating with each other to meet the mission objectives. An efficient coverage path planning (CPP) collision-avoidance capable algorithm for single or multi-UAV systems in cluttered urban environments is presented in Muñoz et al. (2021). The algorithm uses a boustrophedon flight pattern with UAVs flying in a triangular deformable leader–follower formation. This formation can lead to over-covering problems making coverage missions less effective. A prominent approach to UAV flocking as discussed in Vásárhelyi et al. (2014) involves a decentralized solution for multicopter flocks, drawing inspiration from the collective behavior of flocks, herds, and schools (Reynolds 1987). Agents operate autonomously, receiving information from nearby UAVs and employ GPS receivers and wireless interface for stability. Another approach, described in Schleich et al. (2013) employs a leader–follower strategy, assuming constant speed and altitude for UAVs. These UAVs experience stochastic dynamics, and the control model for each UAV is based on stochastic optimal control, factoring in distance and heading angle of the leader. This approach was tested with a group of three camera-equipped UAVs for vision-based target tracking, resulting in noticeable decrease in sensing errors and sensor faults. The work presented in Ruetten et al. (2020) focused on providing self-organizing capability for UAVs to optimize area coverage. The UAVs read the received signal (RSSI) strength and adjust their positions accordingly. A specific strategy to coordinate dynamic sets of UAVs to solve a specific area coverage problem based on coverage maps is proposed in Pellegrino et al. (2020). The work identified the main subproblems of area partitioning, traversal strategy and repartitioning, and described how each of them can be addressed. The process is evaluated based on the time taken to complete coverage of the interest area and by creating motion heat maps as latency in these networks cannot be tolerated during critical missions. By employing a UAV enabled multitarget tracking and sensing framework, the approach proposed in Patrizi et al. (2020) optimizes data collection and processing through intelligent matching and distributed decision-making. This can lead to more efficient and timely acquisition of critical information, thus potentially reducing overall latency in disaster response and surveillance efforts. By introducing fault-tolerant coordination strategies, the framework may help mitigate delays caused by UAV failures or disruptions, further enhancing the overall responsiveness of the system. With a focus on addressing user-centric concerns such as data processing and transfer times in disaster-affected areas, Miyano et al. (2019) introduced a scheduling method for multi-UAV search systems in disaster scenarios through utility-based problem formulation to optimize data collection while minimizing latency. A coverage control method tailored for multiagent systems is introduced in Aminzadeh and Khoshnood (2023), specifically addressing initial disaster assessment and search tasks. However, it does not delve into flight formations or the extensive coverage of disaster areas or the issue of node failures. The study presented in Kumar and Kumar (2023) explores and analyses the existing research on various coverage techniques for UAVs. It provides an overview of the current state-of-the-art CPP methods for UAVs, examines a variety of geometric flight patterns, and also discusses the key challenges and requirements.

A decentralized and online heuristic approach is proposed in Zhu et al. (2021) to address the challenge of collaborative coverage by multiple UAVs. In this approach, each UAV maintains a probability grid map as a locally stored matrix, and there is no need for shared memory among the UAVs. The strategy is designed around two evaluation functions and the corresponding technical methods that allow UAVs to autonomously make decisions in a self-organizing manner. The simulation results demonstrate that this algorithm effectively combines geometric features like parallel search and internal spiral search for detecting targets in the given area. The scheme discussed in Huang et al. (2021) uses UAVs with image processing to monitor congested roads in various modes. Communication between UAVs is limited, facilitating real-time operation, but it does not explicitly address node failures. Addressing node failures would require additional measures such as redundancy in communication pathways or fault-tolerant algorithms. The algorithm presented in Tnunay et al. (2021) makes use of a virtual leader network topology to guide the formation of UAVs and calculates optimal trajectories for CPP. These trajectories are specifically designed to accommodate the dynamic limitations of the UAVs and takes into account constraints related to the position, velocity, and acceleration. Furthermore, the algorithm addresses the dynamics of the formation itself and employs a feedback controller to adapt and fine-tune the parameters in real-time. The study in Li et al. (2022) provides a more comprehensive understanding of the coverage problem of UAV networks by classifying them into static or dynamic coverage, autonomous or inter-user coverage, homogeneous or heterogeneous coverage, and various other constraints such as energy, obstacles, connectivity, and threats. A mission-based UAV swarms coordination approach discussed in Fabra et al. (2020) uses a centralized approach where the master UAV synchronizes all slave UAVs each time they reach an intermediate point in the mission. This adds control overhead in the network and introduces delay in mission completion. The work in De Benedetti et al. (2017) analyses the self-organizing capability of flock on a monitoring mission. The study examines how a single fault agent can affect the mission performance, in terms of mission time alone. Nonetheless, the paper has contributed significantly in understanding the theoretical aspects of flock formation and the principles of mission planning. A mission-oriented framework to support the management and collaboration of both aerial (UAVs) and terrestrial (UGVs) robots, equipped with sensors and ac-

Fig. 1. Network scenario considered in the work.



tuators is investigated in Pace et al. (2016). This framework allows to manage UAV swarms based on mission-oriented objectives. A series of experiments using off-the-shelf UAVs in a real test-bed was conducted to analyse this aspect. The article presented in Khan et al. (2022) examines the contribution of MASs in disaster recovery missions, particularly highlighting recent research employing UAVs in disaster management. It underscores the importance of tackling coverage control challenges and ensuring fault tolerance by proposing a robust heterogeneous multi-agent approach. However, the focus remains on UAVs stationed at specific locations, neglecting to address node failures in aerial surveillance missions where UAVs cover broader regions and may encounter operational failures.

3. Mission model and environment

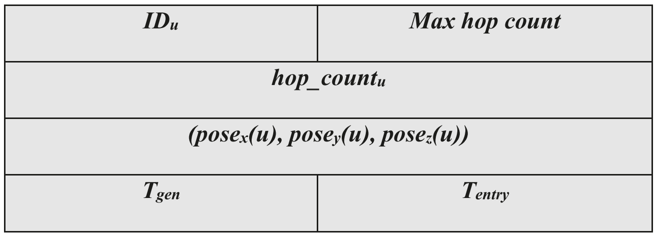
In the context of an aerial survey mission, the objective of this study is to monitor a specific geographical area with a set of UAVs. The study assumes a homogeneous set; however, the analysis may hold good for heterogeneous sets as well since the architecture is decentralized and task allocation considers the capabilities of each UAV. The survey missions can encompass various scenarios, such as conducting aerial photogrammetry of a region (Watanabe and Kawahara 2016; Jiménez-Jiménez et al. 2021), performing aerial inspections to identify specific features or terrain conditions (e.g., landslides), or creating thermographic maps through radiometric infrared cameras. In all these scenarios, it is assumed that the UAVs are equipped with an optical sensor capable of capturing an aerial image of a specific region of the terrain from a certain altitude. The captured area at an instant is referred to as the camera footprint and is taken as a rectangular region with dimensions (w, l) that depends on the altitude and sensor characteristics; w and l represent the breadth and length of the footprint, respectively, at an altitude h relative to the ground level. The area to be surveyed is assumed to exhibit a rectangular shape with dimensions $(W \times L)$, where W is the width and L is the length of the region. This assumption does not limit our approach, as it remains feasible to identify the smallest enclosing rectangle even in scenarios where the actual area exhibits a different geometric configuration. Within this environment, the monitoring entities, hereafter referred to as agents, consist of multirotor vehicles with four-degrees-of-freedom and supports vertical take-off and landing. Additionally, the area to be monitored includes a ground control station (GCS) responsible for collecting the data acquired by the agents. Each agent has control over its altitude and the Euler angles roll, pitch, and yaw. According to the dynamics of the multirotor, a non-zero roll or pitch angle leads to a translated flight along the y or x-axis, respectively. As a result, the position *pose* of each agent u at time t is described by a four-element tuple as eq. 1.

$$\text{pose}_u(t) = (x_u, y_u, z_u, \theta_u) \tag{1}$$

where θ_u corresponds to the yaw angle with respect to magnetic North. UAVs are equipped with a flight stack (e.g., Pixhawk autopilot (Mardiyanto et al. 2019) and a range of sensors, including accelerometers, gyroscopes, magnetometers, barometers, and GPS. This ensures that each agent constantly knows its pose at any given instant. Furthermore, each agent is equipped with two communication systems: (i) a low-power, short-range system for agent-to-agent communication, (ii) a medium-power, long-range system for communication with the GCS. This dual communication setup is designed to manage power consumption efficiently. While agent interactions are frequent, the long-range system is primarily employed for communication with the GCS. Every agent in the network is identified through a unique ID which remains constant throughout the mission. The overall system architecture along with an illustration of the described scenario, is depicted in Fig. 1.

For a team of UAVs to survey an area effectively, they must collaborate with each other to perform a coordinated flight. A decentralized framework allows each UAV in the team to act autonomously and make decisions based on local information shared between the peers, such as their current position, velocity, sensor readings, and planned flight paths. This can be achieved through short range wireless communication protocols such as Wi-Fi, bluetooth, or specialized protocols designed for UAV networks (Chen et al. 2020). Although,

Fig. 2. Agent entry in the local database.



ID_u	*Max hop count*
hop_count_u	
($pose_x(u)$, $pose_y(u)$, $pose_z(u)$)	
T_{gen}	*T_{entry}*

in a decentralized framework, there may not be a designated leader responsible for coordinating the entire team, UAVs may employ leader-follower dynamics, where one UAV takes the lead and others follow its trajectory while maintaining the formation. While the leader UAV provides guidance, decision-making remains decentralized, with each UAV capable of adjusting its actions based on local observations and communication with neighboring UAVs. This allows the team to respond effectively to dynamic environments and uncertainties. When employing a line formation with UAVs, it is possible that the communication range of individual UAVs may not cover the entire group, especially if the formation spans a large area. In such cases, a multihop forwarding mechanism becomes necessary to ensure effective communication and coordination across the entire network (Bekmezci et al. 2013). This approach ensures that every agent is aware of every other agent in the network along with their relative positions. Each agent maintains a neighbor entry table in its local database. For each agent u, the information stored are, the agent ID_u, the last known positions ($\mathrm{pose}_x(u)$, $\mathrm{pose}_y(u)$, $\mathrm{pose}_z(u)$), estimated number of hops i.e., hop count hc_u, local timestamp T_{gen} at which position information is generated, and the time stamp T_{entry} at which the information has been received and entered into the neighbor entry table. The agent entry format is shown in Fig. 2.

When data tuples are broadcasted, any agent within the communication range may receive it. The receiving agent then checks its local database for associated information based on the received ID. If no information is found, the tuple is added to its database. However, if associated information is located, the timestamps of both the received data and the existing entry are compared, and the entry is updated only if the received timestamp is greater than the current entry timestamp; otherwise, it is discarded. This process ensures that received information is relatively fresh, and T_{gen} and T_{entry} are updated in the neighbor table entry. This data exchange is done periodically to maintain the formation throughout the mission. Each entry in the neighbor table has a lifetime, denoted as *MAX_PERIOD*. An entry that is not updated within this period is considered stale and is removed. This may occur if an agent fails and its data are no longer transmitted over the network. Setting an appropriate value for *MAX_PERIOD* is essential to ensure that the lack of data updates does not result from packet loss due to collisions or network congestion. The Max hop count field decides the distance the packet can travel and is set to the overall diameter (*net_diameter*) of the network.

A. Flock formation

After an initial setup time, $T_{\text{set-up}}$, the agents are assumed to have knowledge of other agents and their distances in terms of hop count. For time t greater than $T_{\text{set-up}}$, the agents utilize formation rules to establish a proper flock. The next fundamental requirement is to achieve an optimal flight path and an ideal flock shape that prevents repeated coverage of the same region by one or more UAVs, referred to as overcovering, as it is essential for the chosen application. Hence, it is logical to have UAVs fly side-by-side to prevent one agent from flying behind another, as other formations can add to over coverage problem (De Benedetti et al. 2017). Any flock formation that adheres to this aspect, such as a V-shape or a linear pattern aligns with our application requirements.

Typically, flock formation approaches, as discussed in studies (Reynolds 1987; Bouraqadi et al. 2009; Vásárhelyi et al. 2014), rely on algorithms that employ information on agent positions and apply the following rules:

- Separation (R1): Enforces a minimal distance to avoid crowding and potential collisions.
- Alignment (R2): Ensures that all agents maintain the same heading and direction of movement.
- Cohesion (R3): Aims to maintain an interconnected cluster, preventing agents from drifting apart and causing a potential flock partition.

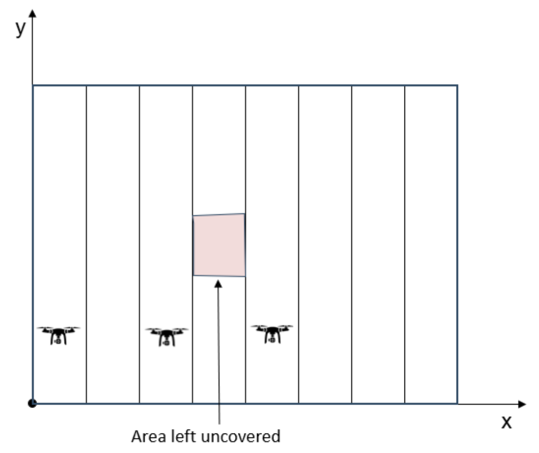
Based on the preceding considerations regarding flock shape, we have chosen the linear placement of agents along a formation line perpendicular to the flight direction. In this formation, the distance between two adjacent agents should ideally match the width of the sensor view w at the flight altitude h. This choice, implemented by configuring parameters within rule R1, helps fulfil both the previously mentioned requirements.

A virtual leader agent is selected by identifying the agent with the lowest *ID* among those in the database. This *ID* is stored as *currentLeader_ID* in the local database of each agent. With every agent now informed about its peers, they all choose the same leader. Although a leader is selected, decision-making remains decentralized, with each UAV capable of planning its actions based on local observations and communication with neighboring UAVs. The leader acts as a relay between the GCS and the follower UAVs, transmitting data collected by the UAVs to the GCS and vice versa. With the leader in place, a virtual line perpendicular to its heading direction is taken as the formation line, as shown in Fig. 1. To establish the flock, all agents strive to reach and maintain positions on this formation line while adhering to rules R1, R2, and R3.

B. Area coverage and path planning

The process of area coverage is a collaborative task involving all UAVs. It relies on the continuous exchange of data related to the area already covered by various agents and on the planning of an optimal path for the flock to survey the remaining portions. The leader holds the responsibility

Fig. 3. Area subdivision in the proposed work.



of relaying the waypoints; and the GCS located at considerable distance from the surveyed area serves as an additional entity for data gathering regarding the covered area. Therefore, there are two crucial mechanisms: (i) agents establishing and maintaining a local database for the covered area parts, known as the covered area database (CAD); and (ii) the leader periodically executing a querying mechanism to collect updates on coverage from the other UAVs, merging them, and the swarm devising an appropriate survey path.

In a line formation, dividing the area into segments is a common strategy to efficiently cover the entire region while ensuring that each segment is adequately surveyed (Muñoz et al. 2021). The area to be surveyed is divided into a series of contiguous stripes, typically arranged linearly along the desired direction of movement for the UAV swarms as depicted in Fig. 3. The number and size of stripes may vary depending on factors such as the area, complexity, and the capabilities of the UAVs. Each stripe identified for surveying is then further subdivided into smaller segments based on the size and shape of the camera footprint. The camera footprint of each UAV is determined based on camera specifications, such as field of view, resolution, and altitude; and can be calculated from eq. 2 and eq. 3, where r is the camera aspect ratio. Within each segment, specific waypoints (geometric center of the footprint area) are extracted to guide the UAV flight path during the surveying mission. Further, the UAVs autonomously determine its own set of waypoints and collaborates with neighboring UAVs to ensure comprehensive coverage of the segmented area.

$$w = 2htan\left(\frac{\theta}{2}\right) \tag{2}$$

$$l = \frac{w}{r} \tag{3}$$

To simplify the representation of the area to be covered, as well as the individual portions that can be monitored by a sensor, we shift from a global geographic reference where geographic coordinates, namely latitude, longitude, and altitude, are mapped to a local cartesian reference xyz. One of the corners of the rectangular area becomes the origin of the local xy Cartesian system, while the mission altitude along the z-axis is set as a constant determined by the mission. The formation line of the flock is parallel to the x-axis, and the flight is executed with UAVs heading perpendicular to this direction. Since the camera footprint can be known in advance, the area subdivision can be performed before start of the mission. Each subdivision or the stripe (along the x-axis) is denoted by a *stripe_ID*, coordinates (x_{start}, y), and (x_{end}, y) where x is defined in eq. 4 as

$$x = \sum_{i=1}^{n} (x_{\text{end}} - x_{\text{start}}) \tag{4}$$

where n represents the number of stripes, each having width equal to the camera footprint. The segments are represented by *segment_ID*.

Every agent maintains a CAD which contains data representing the strip segments already covered by that specific agent. An entry in the CAD includes {*stripe_ID*, *segment_ID*}. The base station also maintains a base station CAD (BCAD), which follows the same structure as the CAD. The BCAD keeps track of the regions for which monitored data has been received by the base station.

Fig. 4. Query packet format.

Seq_No	*leader_ID*
ack_field	
lifetime	
{CAD}	
mission_terminate	

1. Distributed data aggregation (DDA)

The virtual leader is responsible for gathering information on the covered segments from the follower UAVs and forward it to the GCS. It periodically sends out a query to collect information stored at the local database of each agent. The query data packet format is shown in Fig. 4 and it consists of the following, (i) ID of the leader (*leader_ID*); (ii) sequence number (*Seq_No*) which is incremented by 1 for each new query; (iii) acknowledgement field (*ack_field*) which is a bitmap of size N, each bit (referred to as *ack_bit*) if set, indicates that the corresponding agent has answered to the query; where N is the number of agents in the network; (iv) lifetime field (*lifetime*) which indicates the time for which the packet traverses through the network; *lifetime* equals the network diameter (*net_diameter*) if the packet is broadcasted across the network; and (v) the CAD data about the area already covered by the agent.

The leader sets the *leader_ID* field to its own ID, increments the *Seq_No* by 1, sets its corresponding *ack_bit* to 1, and lifetime to *net_diameter.* The query is then broadcasted over the low-power wireless range. The agents which are in range receives the packet and performs algorithm I.

Lines 1. Define the aggregation function with *leader_ID, Seq_No, ack_bit,* and *lifetime* as input parameters.

Lines 2–4. Check if *mission_terminate* flag is set: if yes, terminate the mission; otherwise, check if *leader_ID* is same as *currentLeader_ID,* and if *Seq_No* > *currentSeq_No*; if conditions are met, proceed to update the packet with its partial CAD information; the packet is dropped otherwise. A packet mismatch could happen when the leader fails and the information on current leader is not updated. The *Seq_No* is checked to avoid stale data in the network.

Line 5. *currentSeq_No* updated.

Lines 6–7. The *CAD* information in the received packet is merged with the CAD in the agent database and the CAD field in the packet is replace with the merged data. The merge process is described in the algorithm and involves union operation of the stripe segments and eliminating the overlapped segments. The *ack_bit* field corresponding to the agent is set.

Lines 8. The packet is further broadcast to other agents in range.

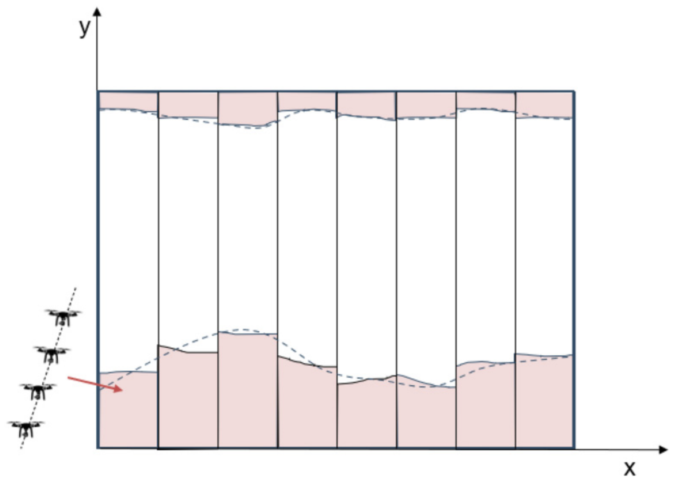
When the packet returns to the leader with the *ack_field* set to 1, the CAD information received can be considered as completed. This information is sent to the base station to keep a track of the mission execution. The *mission_terminate* flag is set when all the stripes are covered by the agents.

Algorithm I. Distributed data aggregation.

```
1:  function CAD_aggregate (receivedCAD (leader_ID, Seq_No,
    ack_bit, lifetime)
2:  if leader_ID = currentLeader_ID do:
3:  if (not mission_terminate) then
3:  if Seq_No > currentSeq_No do:
4:  currentSeq_No ← Seq_No
5:  Set ack_bit (agent)
6:  CAD ← merge(ReceivedCAD, partialCAD (agent))
7:  Broadcast packet to neighbors
8:  else
9:  Discard the packet
    else
    Stop mission and land
10: else
11: Discard the packet
```

Following the exchange and merging of CAD data, the leader proceeds to determine the uncovered waypoints and further computes the possible paths by considering the external borders of the uncovered segments. It is assumed that sufficient number of agents have the energy to traverse the uncovered path. Agents with energy less than a threshold informs its neighbors and proceeds to land at the depot. Due to the coordinated movement of the agents, they usually focus on a single unexplored region at a time. Hence, considering the external borders of this single unexplored area can result in two potential paths, depending on whether the flock starts tracing the area boundary clockwise or counterclockwise. The leader evaluates these two paths and selects the one that is closer to the flock. If there are multiple disjoint uncovered segments, the leader strategically selects the optimal path by considering the size of the areas to be covered and their proximity to the flock. This strategy is designed to maintain the efficiency of the flock in covering as many of the uncovered regions of interest as possible.

Fig. 5. Possible path followed by the leader.



Once the area is identified, the region boundaries are smoothened as shown in Fig. 5 and the smallest rectangle that fits the smoothened area is obtained to further extract new set of waypoints. The leader UAV then broadcasts the extracted waypoints to all agents within the swarm. This communication ensures that all UAVs have access to the same waypoints and can autonomously plan their flight paths accordingly. Each UAV autonomously selects the waypoints that it can cover based on factors such as remaining energy levels and proximity to the waypoints. This decision-making process allows UAVs to optimize their flight paths based on their individual capabilities and constraints. By sharing information about selected waypoints and coordinating their movements, UAVs can avoid redundancies and gaps in coverage, maximizing the efficiency of the surveying mission. When the leader identifies that the whole area has been covered, it broadcasts a *mission_terminate* flag to signal other agents to stop following the formation, fly back to the depot and eventually land.

Thus, even if an agent encounters issues or fails during the operation the coverage objective is met by the rest of the flock seamlessly adapting to the mission requirements. However, there are two faulty scenarios to be addressed: (i) non-leader agent failure, and (ii) leader agent failure. When a non-leader agent fails, it stops broadcasting its position and data. The neighbor agents detect that the entry for the faulty agent has not been updated for *MAX_PERIOD* and the entry is removed from the agent CAD. However, the flock can continue its operation by considering only the agents that are still operational. In the next round of the query, the leader identifies that the *ack_bit* field for the fault agent is not set and assumes that the area covered by the faulty agent is not included in the packet; the leader now plans a mission that incorporates the lost data, effectively re-covering it. This new path may lead to some areas being covered more than once, but this is a trade-off for data recovery.

A similar situation arises if the agent at fault is the leader. Consequently, the agent is removed from CAD and a new leader is selected. After *MAX_PERIOD*, the agent is removed from the agent database, and a new leader is selected. However, because there is no global synchronization, not all agents will synchronize the removal of the leader tuple at the same time. This can lead to a temporary interval during which all agents may not agree to a single leader. This, however, does not pose a problem in practice as agents do not follow the leader directly; instead fly autonomously. Thus, they will continue to maintain their formation, and compensates for the latency required to elect a new leader.

Algorithm II. Waypoint selection method.

```
1:  **Inputs**: List of waypoints $\{W_p\}$, current position of $UAV_j$ $(x_j, y_j)$, angle threshold $\alpha_j$, maximum flight duration $D_j$
2:  **Output**: List of selected waypoints for $UAV_j$ to navigate
3:  Initialize: $aligned\_waypoints \leftarrow$ empty list $selected\_waypoints \leftarrow$ empty list
4:  for each waypoint $j$ in $\{W_p\}$ do:
5:  if angle between($UAV_j$, waypoint $j$) < $\alpha_j$ then:
6:  Append $j$ to $aligned\_waypoints$
7:  end if
8:  sort $aligned\_waypoints$ by distance from $UAV_j$
9:  end for
10: $total\_distance \leftarrow 0$
11: for each waypoint in aligned_waypoints do:
12: if ($total\_distance$ + distance between ($UAV_j$, waypoint)) < D then:
13: update UAV position to waypoint
14: $total\_distance \leftarrow total\_distance$ + distance between ($UAV_j$, waypoint)
15: append waypoint to $selected\_waypoints$
16: else
17: break
18: end if
19: end for
20: return $selected\_waypoints$
```

2. Decentralized waypoint selection

The base station transmits the waypoints to all the UAVs through the selected virtual leader. It is to be noted that the leader does not take any decisions on behalf of other UAVs, instead it only relays the information. The UAVs autonomously select the waypoints based on their own capabilities. Algorithm II depicts the process involved in selecting the waypoints by each UAV. The inputs to the algorithm are: list of waypoints, current UAV position, acceptable angle threshold and maximum flight duration of a UAV which is a factor of battery capacity and the discharge rate. Lines 4–7 of the algorithm filters waypoints based on their alignment with the current orientation of the UAV, ensuring that only those within a specified angle threshold are considered. These aligned waypoints are then sorted based on their proximity to the current UAV position as stated in line 8. The algorithm then iterates through the sorted list, evaluating each waypoint selection based on the constraint that adding its distance to the total distance travelled must not exceed the maximum flight duration as depicted in lines 10–12. If a waypoint satisfies this condition, the UAV updates its position accordingly and adds the waypoint to the selected list. The process continues until either the maximum flight duration is reached or there are no more feasible waypoints left. By dynamically updating the UAV path based on proximity and flight duration constraints, the algorithm optimizes navigation efficiency, enabling the UAV to effectively explore its environment while respecting operational limitations.

4. Simulation environment

To evaluate the mission performance and validate the proposed work, we utilized the PX4-SITL simulator in conjunction with the ROS within a 3D Gazebo environment. The network characteristics are analyzed using NS-3. Both the simulation environments are briefed in this section.

A. Robot operating system (ROS)

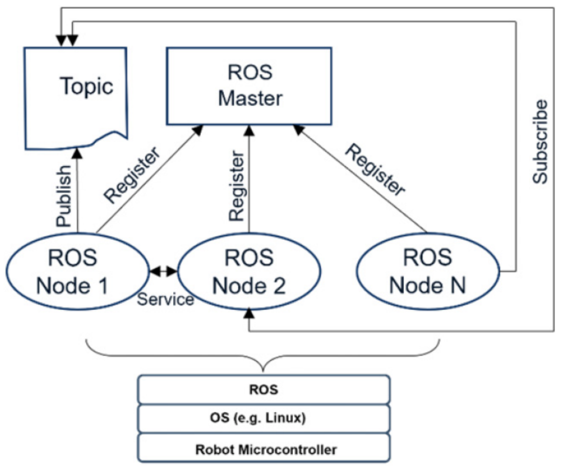
ROS or robot operating system, serves as a middleware that facilitates the seamless operation of diverse devices and systems (Koubâa 2017). At its core, ROS adopts a graph structure where processes are represented as nodes, enabling communication through various messaging patterns like point-to-point and publish/subscribe mechanisms. Central to the ROS network is the master node, which plays a pivotal role in providing essential services such as name registration, query services, and facilitating connections between nodes. Typically, ROS implements the publish/subscribe communication pattern, wherein nodes publish information on specific topics, which is then received by subscribing nodes. This messaging paradigm, coupled with efficient message routing facilitated by naming conventions, forms the backbone of ROS communication. Moreover, ROS introduces the concept of services, enabling one-to-one data exchange between nodes. A node defines a service through which other nodes can interact with it, further enhancing the flexibility and versatility of communication within the ROS ecosystem. The fundamental operation of the ROS is depicted in Fig. 6.

In practical implementations, ROS programs are commonly run on the linux operating system (OS) utilizing the *rospy* library in Python version 3.10. Within the context of UAV flight control, ROS facilitates precise control through the utilization of PID (proportional-integral-derivative) controllers, a widely adopted method in UAV flight stacks. PID control involves fine-tuning parameters related to angular rate, speed, and altitude to achieve desired flight behavior. Communication among agents within the system is facilitated through message exchange using the ROS framework. ROS offers predefined message types for a range of topics, including linear and angular velocities, local and global positions, and it also supports user-defined message types and topics.

B. Network Simulator (NS-3)

The NS-3 simulator stands is an open-source discrete event NS, developed as a platform for networking research. It em-

Fig. 6. Node communication in robot operating system framework.



powers users with a comprehensive suite of models that helps in analysing the performance of packet data transmissions, coupled with a simulation engine that facilitates the execution of simulation experiments. By leveraging NS-3, researchers can delve into evaluations that may be impractical or unfeasible in real-time systems, thus unlocking avenues for in-depth analysis and experimentation. Notably, NS-3 boasts support for both wired and wireless networks, offering a versatile environment for exploring networking protocols across various layers of the OSI model. Built as a modular set of libraries, NS-3 encourages integration not only within its own framework but also with external software libraries, fostering flexibility, and extensibility. This design philosophy enables users to tailor simulations to their specific research needs and integrate additional functionalities seamlessly. Furthermore, NS-3 provides compatibility with a range of external tools dedicated to animation, data analysis, and visualization. For instance, popular tools like *netAnim* for animation and *tracemetrics*, *Wireshark*, and *gnuplot* for data analysis and visualization can be employed in conjunction with NS-3 simulations. However, users typically interact with NS-3 through the command line interface, leveraging C++ and/or Python development tools for scripting and customization. While NS-3 finds its stronghold in Linux and macOS environments, it is worth noting that support for Windows framework is also available, albeit with some considerations (available from https://www.nsnam.org/). This cross-platform compatibility ensures accessibility to a wider user base, fostering collaboration and innovation within the networking research community.

5. Experimental analysis

The experimental analysis for the proposed scheme is performed from two different perspectives. In the first phase, the flock formation and mission performance are analyzed on ROS framework and visualized on Gazebo environment that takes real world physical constraints into consideration. To set up the simulation, we have considered a square area

Fig. 7. Waypoints extracted for an area of 1000 × 1000.

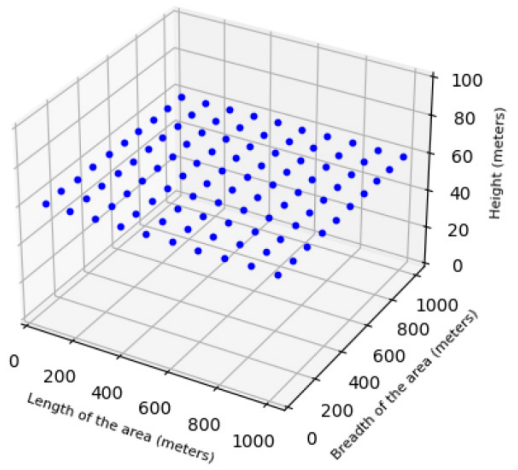
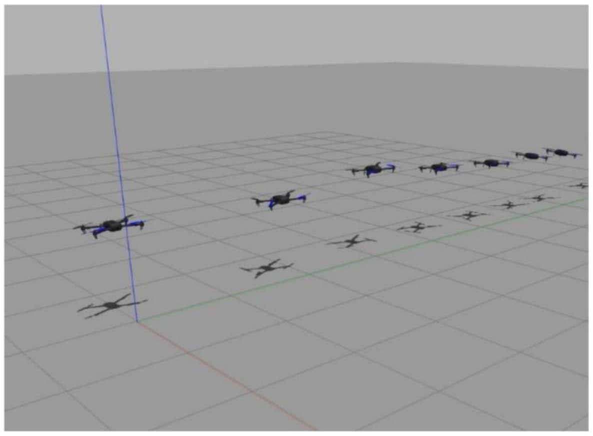


Fig. 8. Agents take-off on robot operating system Gazebo framework.



of 1000 × 1000 m to be covered with quadcopter UAVs. The speed of the nodes is set as 10 m/s, and the flight altitude is assumed at 60 m for which the camera footprint (area captured by the sensor) is 122.18 m (w) × 68.73 m (l) as calculated from eq. 2 and eq. 3. The waypoints for the given area are extracted as depicted in Fig. 7. However, for the flock control algorithm, the parameters are obtained from trial-and-error in which the flock performance is analyzed in terms of collisions and partitions, and then adjusted accordingly. At the start of the simulation, the agents take-off to a predefined altitude and then exchange their position information with their neighboring nodes. Figure 8 illustrates the process of agents initiating take-off within the ROS framework. The flocking process starts with all the agents selecting a node as virtual leader based on metrics such as lowest *ID*, minimum distance to the GCS and residual energy, to relay information between the GCS and the other UAVs. The analysis is carried out for a node count of 10 with node *ID*s from 0 to 9. Assuming the set to be homogeneous, all the nodes select agent 0 as their leader (agent 0 is located close to the base station). Once the leader is selected, it requests the GCS to unicast all the waypoints to it. On receiving the waypoints, the leader ad-

Fig. 9. A plot of mission time versus number of nodes.

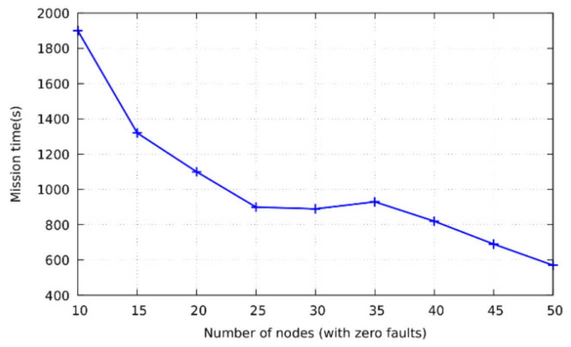


Table 1. Numerical results on mission time, number of nodes, and energy requirements.

No. of agents	Mission time (mm:ss)	Average energy (mAh)	Total energy (mAh)
10	31.67	1005	10 050
15	21.67	918	13 770
20	18.33	841	16 820
25	15.20	765	19 125
30	14.75	623	18 690
35	16.08	779	27 265
40	13.38	502	20 080
45	11.67	472	21 240
50	9.83	458	22 900

justs its heading direction based on the position vector of the waypoint coordinates, while all the follower UAVs align themselves with respect to the leader, perpendicular to its heading direction. Further, the leader broadcasts the waypoints to all the UAVs and each UAV selects its own set of waypoints based on Algorithm II.

The first set of evaluations aim at calculating the mission time versus the number of agents under zero agent failures. Figure 9 shows a plot of mission time versus number of agents, and the numerical values are reported in Table 1. From the results, we can observe that, with a greater number of agents, the mission time is reduced. However, once the number of agents exceed 25, mission time values are relatively stable up to a count of 35 further which the mission time reduces significantly. Figure 10 shows a comparative plot of the decentralized scheme with state-of-art schemes (Fabra et al. 2020) that has considered a centralized coverage, (Schleich et al. 2013) that considered distributed multi-UAV mission and (Aminzadeh and Khoshnood 2023) that has performed cooperative mission for post-disaster scenario. It can be noted that both distributed and decentralized architectures achieve similar mission times for a given number of nodes under the environment considered in the study. However, the decentralized architecture is more reliable during node failures as illustrated in Fig. 11. This is because, each node in the network typically has some degree of autonomy and can make decisions based on local information and interactions with nearby nodes. This ensures that if one node

Fig. 10. A comparison plot of mission time with state-of-art schemes.

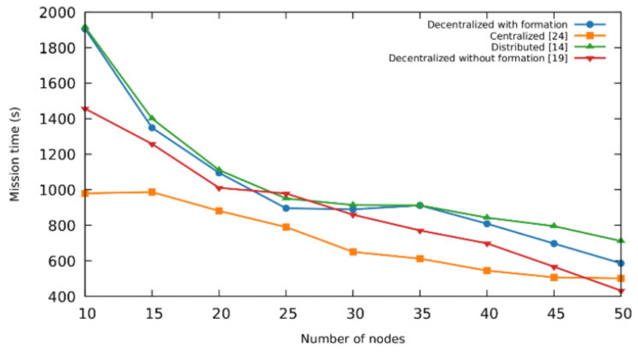


Fig. 11. Reliability versus number of nodes (with single fault).

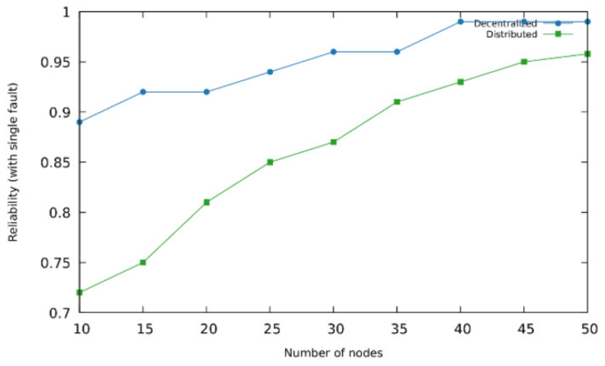
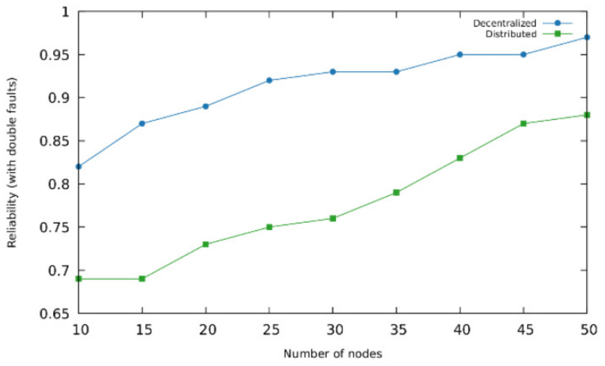


Fig. 12. Reliability versus number of nodes (with double faults).



fails, others can continue to operate and maintain network connectivity. In the analyzed scenario considering both single and double faults, decentralized architectures demonstrate superiority over distributed networks, even though reliability decreases with an increasing number of faults as noted in Figs. 11 and 12. It is to be noted that the study defines node failure as instances where a node is no longer a part of the network, encompassing scenarios such as battery depletion, collisions with obstacles, and environmental uncertainties that render nodes unavailable for network operations. However, it does not classify temporary communication loss as a node failure. Instead, the assumption is made that nodes will regain connectivity and resume their mission after such interruptions. The flocking time is also plotted for different

Fig. 13. Flocking time versus number of nodes.

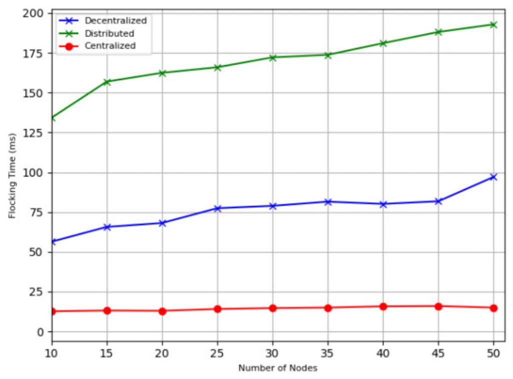


Fig. 14. A plot of area versus coverage times.

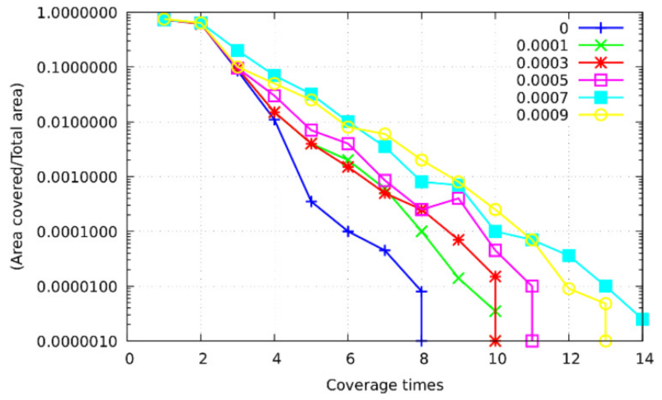
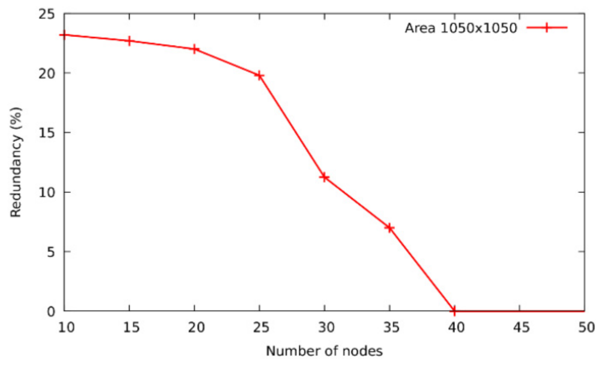


Fig. 15. A plot of redundancy versus number of nodes.



UAV network architectures as depicted in Fig. 13. It can be observed that the flocking time is higher for decentralized scheme. This is because agents within the decentralized environment communicate with neighboring nodes to achieve alignment in formation, contrasting with the distributed environment where nodes receive location coordinates directly from the leader, and from the ground station in case of centralized architecture.

The next objective is to verify the efficiency of the UAV swarms to re-cover the area during agent failures. A plot of recovered area versus single agent failure probability is presented in Fig. 14. We can observe that, average over-covering increases as the failure probability increases. With zero failure probability, more than 70% of the area is covered only once, 20% of the area is covered twice, more than 8% of the area is covered thrice, and <2% covered four times. Though average over-covering increases with failure probability, the numerical values are negligible beyond a coverage of three times. Thus, the propose strategy can be considered effective during single agent failures and relative effective during double node failures. We have also studied the performance of the proposed scheme for multiagent failures, but the coverage efficiency was reduced by 9.2%. Since the application under study is disaster monitoring which is time-sensitive, we manually added redundant agents in the network to analyze what percentage of redundancy helps in achieving the mission in the same amount of time without failures. The total required mission time is assumed to be 35 min for a set of 10 waypoints for each UAV, and two faults are assumed at waypoint 3 (for agent 2) and at waypoint 8 (for agent 7) respectively. With multiple trial-and-error attempts, we have observed a redundancy requirement of 22.7% for node count of less than 25. But when the count increases, this redundancy percent falls, as shown in Fig. 15. This could be because the tasks assigned to the faulty agents are evenly distributed across the other agents in the network.

To compare our work with Pace et al. (2016), we also carried out sub-area coverage for the given area, and the comparison results are shown in Fig. 16. In sub-area coverage, the total area is divided into smaller sub-regions and each region is assigned to a single UAV. With number of agents less than 30,

Fig. 16. Number of nodes versus mission time.

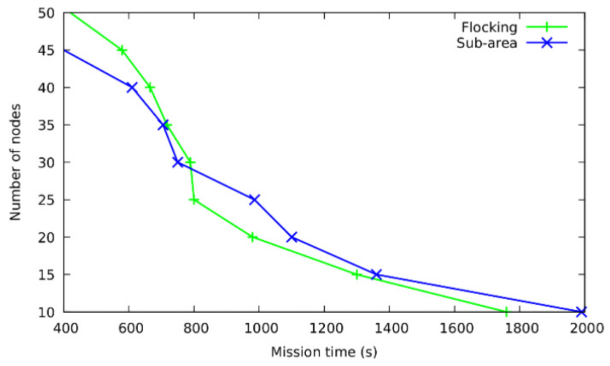
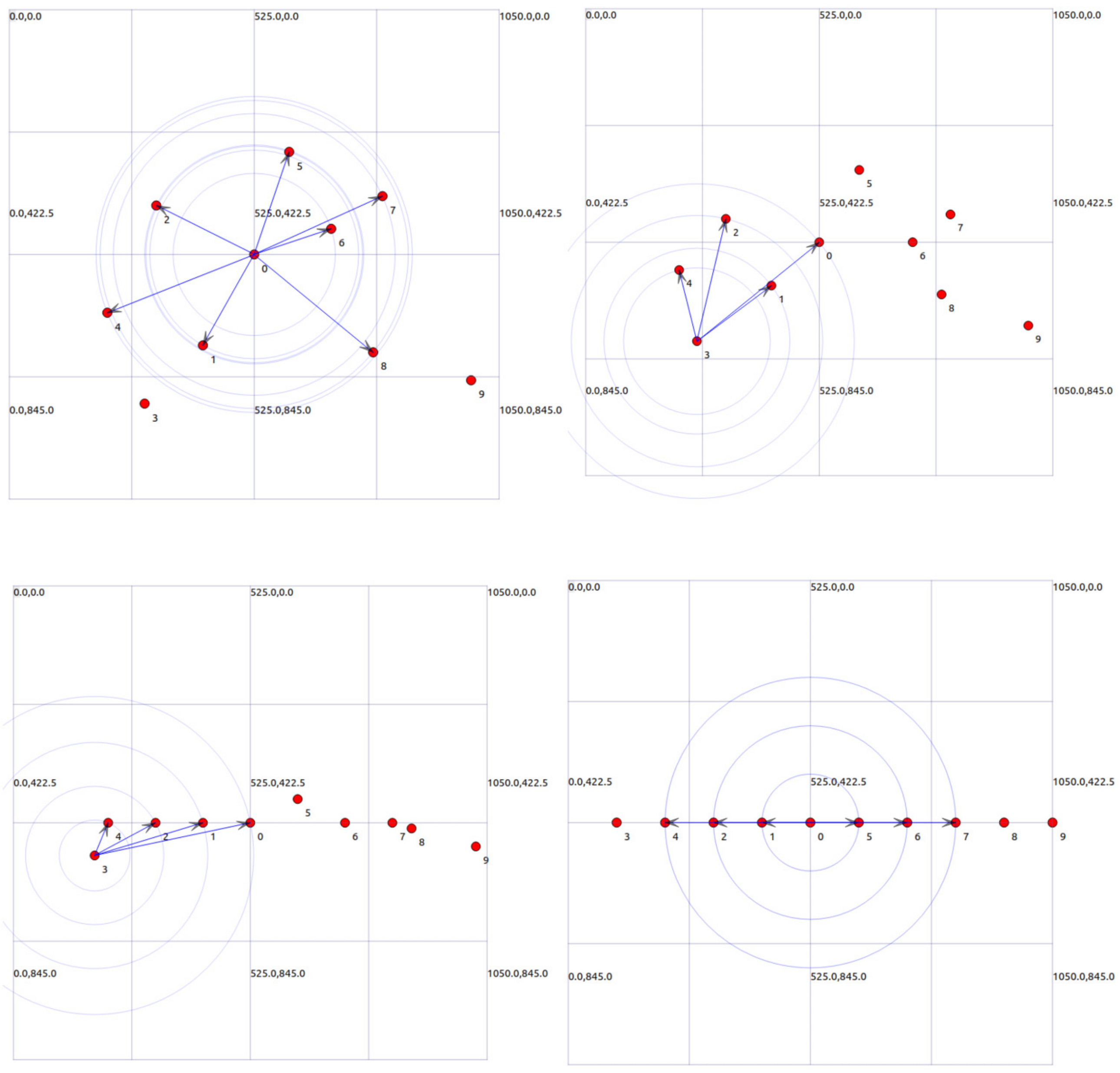


Table 2. Number of agents required and total energy spent versus mission time.

Parameter	Value
Simulation area *(Rect_size)*	1050 × 1050
Simulation time	250 s
Number of nodes	10–50
Initial node energy	100 Joules
Node speed	10 m/s
Mobility model	*Waypoint Mobility Model*
Propagation model	Free space
MAC protocol	IEEE 802.11
Routing protocol	AODV
Physical layer	IEEE 802.11b

Fig. 17. Flock formation process on NS-3 simulator: (*a*) nodes deployed randomly; (*b*) nodes communicate with each other to align themselves in relative positions; (*c*) Nodes aligned partially; (*d*) all nodes in linear formation and the master broadcasts packets.



the sub-area coverage approach proved to be efficient than the proposed strategy. Beyond 30, the proposed strategy completed the mission in lesser time. This could be because of the additional distance that a UAV has to cover from its own assigned area to the sub-area of the UAV that failed. This completes the first phase of evaluation.

The second phase performs protocol stack analysis using NS-3 to study the network performance in terms of network aspects such as end-to-end delay, and packet delivery ratio and node survival rate. The stack design considerations are shown in Table 2. The UAVs are assumed as nodes, and all the nodes in the network are randomly deployed. They exchange *hello* messages to inform other nodes about its presence. That way, each node updates its neighborhood information in the routing table. Since the nodes are battery powered and the network is highly dynamic in nature, energy-aware ad-hoc on-demand routing protocol (AODV) is considered for routing the packets from the source node to the destination node. The flock formation of nodes is tested on the simulator and the results for a total of 50 nodes are discussed in this section. The nodes are placed randomly at the start of the simulation as shown in Fig. 17*a*, and all the nodes communicate with their neighbors to calculate their relative positions as shown in Figs. 17*a* and 17*b*. In Fig. 17*c*, the nodes are partially aligned, with node 0 as the reference leader; the remaining nodes align in linear formation as shown in Fig. 17*d*. The flock formation time is calculated to be 25 s. On comparing the communication overhead in the network during flock

Fig. 18. Communication overhead versus number of nodes.

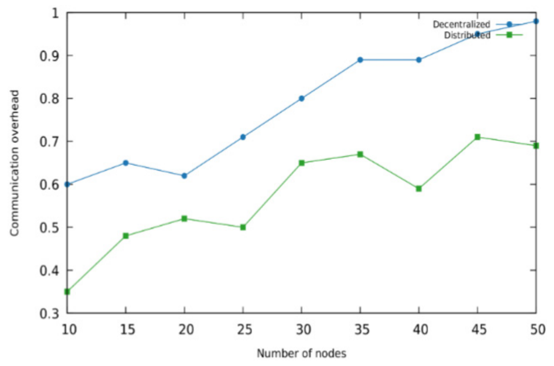
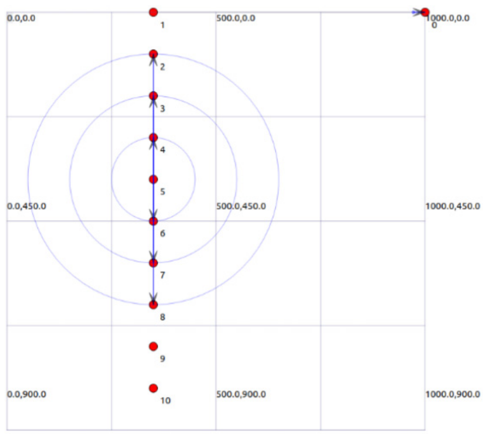


Fig. 19. Network performance of agents in NS-3 framework.



formation for decentralized and distributed architectures, it is observed that the communication overhead is higher for the decentralized scheme as depicted in Fig. 18. Unlike distributed architectures, where decision-making may be more centralized, the decentralized approach distributes decision-making across the network, leading to increased communication requirements among nodes. As a result, while decentralized architectures offer advantages in terms of autonomy and adaptability, they may incur higher communication overhead, particularly in scenarios under study requiring complex coordination and decision-making. Fig. 19 depicts the network environment with ten nodes actively participating in the mission. Subsequently, various network parameters are evaluated. In all the tests conducted, the node speed is set as 10 m/s and the simulation time is 250 s. To analyze the influence of node density on network performance, the number of nodes is increased from 10 to 50. The numerical values presented are an average of 10 runs. Figure 20 shows a plot of average end-to-end delay for different node densities. It can be observed that the delay is higher for a node count of 30, and decreases as the nodes increases.

This is because, with less number of nodes, there may be fewer links connecting the source to destination. As the node count increases, more links are available and the data can be delivered faster. However, from Fig. 21, it can also be observed that, as the nodes further increases, the more and

Fig. 20. A plot average end-to-end delay versus number of nodes.

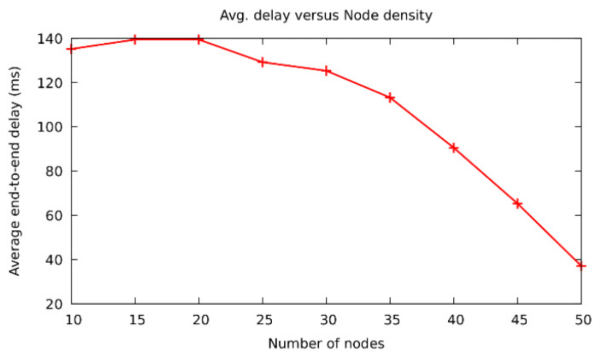


Fig. 21. A plot of routing overhead versus simulation time.

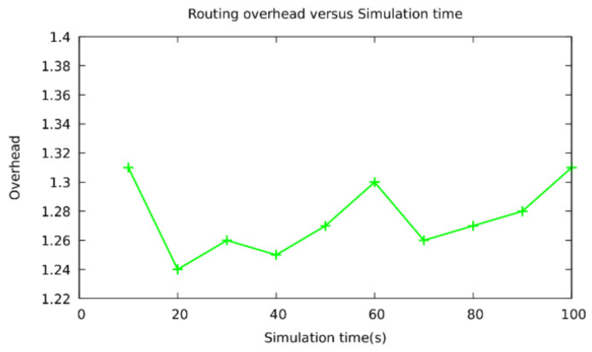
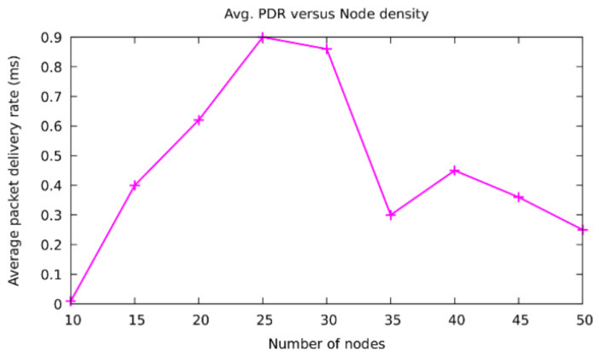


Fig. 22. Average packet delivery ratio versus number of nodes.

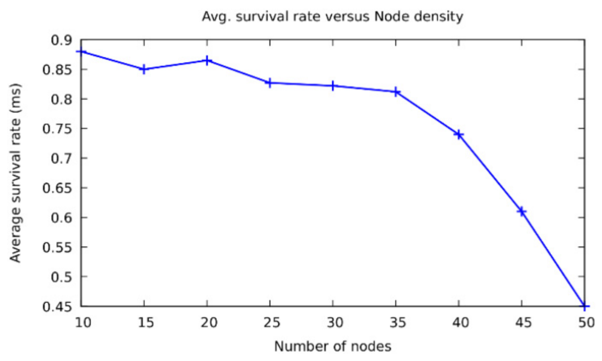


more control packets are broadcasted and this can lead to network congestion. Figure 22 shows a plot of packet delivery ratio versus the number of nodes. From Fig. 23, it can be noted that the survival rate rapidly decreases after a node count of 35. After examining the potential reasons, we arrived at a conclusion that AODV-based protocols consume energy at faster rate with a higher node density.

A. Challenges of the proposed strategy

Despite the benefits, there are certain challenges identified with the proposed strategy that require further investigation and refinement. One of the primary concerns is the communication overhead associated with maintaining constant,

Fig. 23. Average survival rate versus number of nodes.



decentralized communication among UAVs as illustrated in Fig. 18, which can introduce latency and reduce the system responsiveness. Future research could aim at minimizing the communication overhead, thus making decentralized missions a more viable option for dynamic environments. Additionally, the study is performed for a swarm size of 10, however, as the swarm size increases, the complexity of coordination may escalate, potentially impacting the scalability of our approach in a practical scenario. Energy management remains another critical area, where optimizing the balance between operational duration and mission effectiveness is crucial, especially given the varying conditions and unexpected challenges encountered during missions. Furthermore, while our strategy exhibits robust fault tolerance in simulations, the performance under practical scenarios need to be tested for recovery from multiple simultaneous UAV failures, particularly involving a re-election of leader UAV, need to be strengthened to ensure uninterrupted mission execution. Further, external environmental factors and potential interference pose unpredictable challenges that could affect the swarm performance. Addressing these challenges will require a multifaceted approach, including the development of more efficient communication protocols, advanced energy management techniques, and scalable coordination algorithms. Future research will focus on enhancing the resilience to operational uncertainties and environmental conditions, thereby solidifying the foundation for deploying UAV swarms in a broader range of critical applications.

6. Conclusions

The study explores decentralized fault-tolerant UAV swarm coordination for critical missions, emphasizing the growing importance of UAVs in disaster response. The scheme allows UAVs to self-organize in a linear formation, find optimum coverage paths, and adapt to agent failures. The core advantages of our approach are manifested in its ability to enhance operational efficiency, improve fault tolerance, and ensure adaptive mission execution without the need for centralized control. By enabling UAVs to self-organize into linear formations and dynamically adapt their flight paths, the mission time is significantly optimized, thereby increasing the overall effectiveness. The decentralized nature of the strategy ensures that the system is resilient to single points of failure, a critical advantage in disaster scenarios where reliable performance is paramount. Through testing and validation of the approach on two different platforms, the study validates the effectiveness of the decentralized scheme in achieving higher resilience to node failures. The high level of fault tolerance and efficient flock formation underscores the robustness of the proposed strategy, making it a valuable contribution to the field of UAV networking in diverse applications. While acknowledging the benefits of decentralized architectures and distributed control among multiple UAVs, the study also highlights challenges associated with managing interactions and dependencies among the decentralized nodes. For future work, the authors look forward to address these challenges and in integrating security measures to address vulnerabilities in the network.

Article information

History dates

Received: 14 November 2023
Accepted: 25 April 2024
Accepted manuscript online: 1 May 2024
Version of record online: 4 June 2024

Copyright

Data availability

Data available within the article.

Author information

Author ORCIDs

Indu Chandran https://orcid.org/0000-0002-1374-9737

Author contributions

Conceptualization: IC
Data curation: IC
Formal analysis: IC
Methodology: IC
Resources: IC
Software: IC
Supervision: KV
Validation: IC, KV
Visualization: IC
Writing – original draft: IC
Writing – review & editing: IC, KV

Competing interests

The authors have no competing interests that are relevant to the content of this article.

Funding information

No funding was received to assist with the preparation of this manuscript.

References

Aminzadeh, A., and Khoshnood, A.M. 2023. Multi-UAV cooperative search and coverage control in post-disaster assessment: experimental implementation. Intel. Serv. Robot. **16**(4): 415–430. Available fromdoi:10.1007/s11370-023-00476-4.

Basir, R., Qaisar, S., Ali, M., Chughtai, N.A., Imran, M.A., and Hashmi, A. 2021. Performance analysis of UAV-enabled disaster. Autonomous airborne wireless networks. (Preprint).

Bekmezci, I., Sahingoz, O.K., and Temel, Ş. 2013. Flying ad-hoc networks (FANETs): a survey. Ad Hoc Networks, **11**(3): 1254–1270. Available at:. doi:10.1016/J.ADHOC.2012.12.004.

Bouraqadi, N., Doniec, A., and Douai, E. 2009. Flocking-based multi-robot exploration. National conference on control. Available from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.194.1900&rep=rep1&type=pdf.

Bürkle, A., Segor, F., and Kollmann, M. 2010. Towards autonomous micro UAV swarms. J. Intell. Robot. Syst. **61**(1): 339–353. doi:10.1007/S10846-010-9492-X.

Calafate, C.T., and Tropea, M. 2020. Unmanned aerial vehicles—platforms, applications, security and services. Electronics, **9**(6): 1–3. doi:10.3390/electronics9060975.

Chen, X., Tang, J., and Lao, S. 2020. Review of unmanned aerial vehicle swarm communication architectures and routing protocols. Appl. Sci. **10**(10). doi:10.3390/app10103661.

De Benedetti, M., D'Urso, F., Fortino, G., Messina, F., Pappalardo, G., and Santoro, C. 2017. A fault-tolerant self-organizing flocking approach for UAV aerial survey. J. Netw. Comput. Appl. **96**(August): 14–30. doi:10.1016/j.jnca.2017.08.004.

Everaerts, J. 2014. The use of unmanned aerial vehicles (UAVs) for remote sensing and mapping. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciendes. **37**(Inter-Commission WG I/V): 1187–1191.

Fabra, F., Zamora, W., Reyes, P., Sanguesa, J.A., Calafat, C.T., Cano, J.-C., et al. 2020. MUSCOP: mission-based UAV swarm coordination protocol. IEEE Access, **8**: 72498–72511. doi:10.1109/ACCESS.2020.2987983.

Huang, H., Savkin, A.V., and Huang, C. 2021. Decentralized autonomous navigation of a UAV network for road traffic monitoring. IEEE Trans. Aerosp. Electron. Syst. **57**(4): 2558–2564. doi:10.1109/TAES.2021.3053115.

Jiménez-Jiménez, S.I., Ojeda-Bustamante, W., de Jesús Marcial-Pablo, M., and Enciso, J. 2021. Digital terrain models generated with low-cost UAV photogrammetry: methodology and accuracy. Int. J. Geo-Inf. **10**(5). doi:10.3390/ijgi10050285.

Khan, A., Gupta, S., and Gupta, S.K. 2022. Cooperative control between multi-UAVs for maximum coverage in disaster management : review and proposed model. 2022 2nd International Conference on Computing and Information Technology (ICCIT). pp. 271–277. doi:10.1109/ICCIT52419.2022.9711627.

Koubâa, A. ed. 2017. Robot Operating System (ROS) Vol. **1**. Springer, Cham, Switzerland.

Kumar, K., and Kumar, N. 2023. Region coverage-aware path planning for unmanned aerial vehicles: a systematic review. Phys. Commun. **59**: 102073. doi:10.1016/j.phycom.2023.102073.

Li, Z., Yan, Y., Xia, X., Xia, L., Meng, D., Chen, C., et al. 2022. A survey of coverage issues in UAV networks. International Conference on Advanced Communication Technology, ICACT, 2022-February. pp. 185–190. doi:10.23919/ICACT53585.2022.9728976.

Luo, C., Miao, W., Ullah, H., McClean, S., Parr, G., and Min, G. 2019. Unmanned aerial vehicles for disaster management. pp. 83–107. doi:10.1007/978-981-13-0992-2_7.

Mardiyanto, R., Pujiantara, M., Suryoatmojo, H., Dikairono, R., and Irfansy, A.N., 2019. Development of unmanned aerial vehicle (UAV) for dropping object accurately based on global positioning system. Proceedings - 2019 International Seminar on Intelligent Technology and Its Application, ISITIA 2019. pp. 86–90. doi:10.1109/ISITIA.2019.8937269.

Miyano, K., Shinkuma, R., Mandayam, N.B., Sato, T., and Oki, E. 2019. Utility based scheduling for multi-UAV search systems in disaster-hit areas. IEEE Access, **7**: 26810–26820. doi:10.1109/ACCESS.2019.2900865.

Muñoz, J., López, B., Quevedo, F., Monje, C.A., Garrido, S., and Moreno, L.E. 2021. Multi uav coverage path planning in urban environments. Sensors, **21**(21). doi:10.3390/s21217365.

Pace, P., Aloi, G., Caliciuri, G., and Fortino, G. 2016. A mission-oriented coordination framework for teams of mobile aerial and terrestrial smart objects. Mobile Netw. Appl. **21**(4): 708–725. doi:10.1007/s11036-016-0726-4.

Patrizi, N., Fragkos, G., Ortiz, K., Oishi, M., and Tsiropoulou, E.E. 2020. A UAV-enabled dynamic multi-target tracking and sensing framework. Proceedings—IEEE Global Communications Conference, GLOBECOM[Preprint]. doi:10.1109/GLOBECOM42002.2020.9322567.

Pellegrino, G., Mota, G., Assis, F., Gorender, S., and Sá, A. 2020. Simple area coverage by a dynamic set of unmanned aerial vehicles. Brazilian Symposium on Computing System Engineering, SBESC, 2020-November. doi:10.1109/SBESC51047.2020.9277866.

Reynolds, C.W. 1987. Flocks, herds, and schools: a distributed behavioral model. Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987. **21**(4): 25–34. doi:10.1145/37401.37406.

Ruetten, L., Regis, P.A., Feil-Seifer, D., Sengupta, S., et al. 2020. Area-optimized UAV swarm network for search and rescue operations. 2020 10th Annual Computing and Communication Workshop and Conference, CCWC 2020. 613–618. doi:10.1109/CCWC47524.2020.9031197.

Schleich, J., Panchapakesan, A., Danoy, G., and Bouvry, P. 2013. UAV fleet area coverage with network connectivity constraint. MobiWac 2013 - Proceedings of the 11th ACM International Symposium on Mobility Management and Wireless Access, Co-located with ACM MSWiM 2013. pp. 131–138. doi:10.1145/2508222.2508225.

Tnunay, H., Moussa, K., Hably, A., and Marchand, N. 2021. Virtual leader based trajectory generation of UAV formation for visual area coverage. IECON Proceedings (Industrial Electronics Conference. 2021-October. pp. 1–6. doi:10.1109/IECON48115.2021.9589446.

Van Dyke Parunak, H., Brueckner, S.A., and Odell, J.J. 2003. Swarming coordination of multiple UAV's for collaborative sensing. 2nd AIAA 'Unmanned Unlimited' Conference and Workshop and Exhibit. pp. 15–18. doi:10.2514/6.2003-6525.

Vásárhelyi, G., Virágh, C., Somorjai, G., Tarcai, N., Szörényi, T., Nepusz, T., and Vicsek, T. 2014. Outdoor flocking and formation flight with autonomous aerial robots. IEEE International Conference on Intelligent Robots and Systems. pp. (Iros), 3866–3873. doi:10.1109/IROS.2014.6943105.

Wang, Y., Su, Z., Xu, Q., Li, R., Luan, T.H., and Wang, P. 2023. A secure and intelligent data sharing scheme for UAV-assisted disaster rescue. IEEE/ACM Trans. Netw. **31**, 1–17. doi:10.1109/TNET.2022.3226458.

Watanabe, Y., and Kawahara, Y. 2016. UAV photogrammetry for monitoring changes in river topography and vegetation. Proc. Eng. **154**: 317–325. doi:10.1016/j.proeng.2016.07.482.

Wei, Y., Brian Blake, M., and Madey, G.R. 2013. An operation-time simulation framework for UAV swarm configuration and mission planning. Procedia Comput. Sci. **18**: 1949–1958. doi:10.1016/j.procs.2013.05.364.

Zhou, B., Xu, H., and Shen, S. 2023. RACER: rapid collaborative exploration with a decentralized multi-UAV system. IEEE Trans. Robot. **39**(3): 1816–1835. doi:10.1109/TRO.2023.3236945.

Zhu, K., Han, B., and Zhang, T. 2021. Multi-UAV distributed collaborative coverage for target search using heuristic strategy. Guid. Navigat. Control. **01**(1): 1–24. doi:10.1142/S2737480721500023.