Drone Systems and Applications

OPEN ACCESS | Article

Cloud-based mathematical models for self-organizing swarms of UAVs: design and analysis

Suren Poghosyan[a], **Vahagn Poghosyan**[a], **Sergey Abrahamyan**[a], **Artyom Lazyan**[a], **Hrachya Astsatryan**[a], **Yeghisabet Alaverdyan**[a,b], **and Karen Eguiazarian**[c]

[a]Institute for Informatics and Automation Problems of NAS RA, Yerevan, Armenia; [b]EKENG CJSC, Yerevan, Armenia; [c]Tampere University, Tampere, Finland

Corresponding author: **Vahagn Poghosyan** (email: povahagn@gmail.com)

Abstract

Unmanned aerial vehicle (UAV) swarms have gained significant attention for their potential applications in various fields. The effective coordination and control of UAV swarms require the development of robust mathematical models that can capture their complex dynamics. The paper introduces mathematical models and relevant paradigms based on the design and analysis of self-organizing swarms of UAVs. The logical and technological construction of the model relies on the theorems developed by authors for obtaining full information exchange during the swarm quasi-random walk. The suggested rotor-router model interprets the discrete-time walk accompanied by the deterministic evolution of configurations of rotors randomly placed on the vertices of the swarm graph. The recommended optimal and fault-tolerant gossip/broadcast schemes support the resilience of swarm to internal failures and external attacks, and cryptographic protocols approve the security. The proposed cloud network topology serves as the implementation framework for the model, encompassing various connectivity options to ensure the expected behavior of the UAV swarms.

Key words: swarm of UAVs, gossip/broadcast models, rotor-router walk, self-organized criticality, cloud platform

1. Introduction

Unmanned aerial vehicle (UAV) swarms, integral to diverse applications (Zhou et al. 2020), excel in meeting high-demand and mission-specific goals. Their efficiency, flexibility, stability, and minimal resource consumption outperform traditional methods. Tailored for specific missions, UAVs offer distinct features. Advantages of swarms include resilience to disruptions, ease of device removal or replenishment, and accelerated mission performance through parallelized operations. The self-organizing ability of swarms enhances adaptability in dynamic environments.

A comparative analysis of UAVs as aerial robots is available in Tahir et al. (2019), describing key characteristics and application areas. The work also presents the swarms' control mechanisms and management. UAV swarms can be represented as self-organized systems that rely on decentralized decision-making and coordination among agents to operate effectively (Campion et al. 2018). Swarm intelligence leverages advanced computing and artificial intelligence technologies to facilitate efficient communication and coordination among agents, promoting collective decision-making to cope with uncertainty and challenges (Babaoglu et al. 2002). In that context, each UAV in a swarm acts as a separate agent embedded with algorithms addressing individual and group behavior (Sneyd et al. 2001).

Information exchange is critical for effectively operating self-configurable swarms and unified decision-making based on accumulated knowledge. The ability of UAV swarms to accomplish a prescribed mission is influenced by critical factors such as autonomy, built-in awareness, and resiliency in adapting to new actors or losing group members (Bai et al. 2020). These factors highlight the importance of mathematical modeling and swarm intelligence-based optimization algorithms for effective swarm management (Chen et al. 2019).

Furthermore, utilizing cloud technologies, virtual environments, and computing resources is essential to justify the scalability and efficient management of self-organizing UAV swarms. These resources enable the development of optimally distributed software–hardware cloud management systems that deploy multi-agent modeling and swarm intelligence-driven algorithms to enhance the coordination and operation of UAV swarms (Ziquan et al. 2022).

The paper aims to introduce a cloud-based mathematical model designed explicitly for self-organizing UAV swarms. The proposed platform simplifies the deployment of UAV swarms, allowing them to adapt and self-organize in real-time, even when facing changing environmental conditions. The remainder of this paper is organized as follows. Section 2 provides a review of related works in this field. The mathematical models that underlie the construction of self-organizing UAV swarms are described in Section 3, followed

by a detailed presentation of the suggested cloud platform in Section 4. Finally, Section 5 provides the conclusion of this paper.

2. State of the art and related work

The design and operational management of UAV swarms is a rapidly evolving field. The large spectrum of potential applications of multi-agent robotic systems motivates introducing new approaches and solutions, each applicable to performing specific mission-oriented tasks. Examining the construction of swarm intelligence reveals that the development of practical self-organizing systems for real-world applications benefits from incorporating robust and efficient solutions from closely related fields while exhibiting peculiarity in approaches and deployment. However, independent of swarms' mission, the methodology for constructing swarms suggests fulfilling mandatory requirements. These include multi-agency collaboration, self-organization, control-based strategies, resilience against physical or logical disruptions, secure cloud deployment for swarms, equitable resource distribution, verification of UAV membership, and the development of mathematical models underpinning these concepts.

In a notable study by Bellur and Narendra (2006), peer-to-peer (P2P) systems are defined as decentralized, large-scale, and highly dynamic, similar to complex adaptive systems (CAS) in biological and social sciences. The authors propose a P2P solution based on multi-agent and evolutionary programming borrowed from CAS, in which adaptive agents traverse the network, interact with nodes, and cooperate to solve complex problems.

In Picard and Glize (2006), the authors propose a cooperative self-organization approach for artificial systems, where cooperative agents can modify their interactions autonomously. The authors stress the importance of behavioral specifications and a middleware layer that dynamically binds system components at runtime, thus ensuring necessary functionality and dynamic reconfiguration of the system.

The studies (Engelbrecht 2006) introduce mathematical models and new computational paradigms of swarm intelligence, covering function optimization, optimal route finding, scheduling, structural optimization, and data analysis. In Abdenebaoui et al. (2015), the authors model UAV swarms with decentralized processing and control. Here, the processes run simultaneously and autonomously with proper coordination. The proposed model is proved to be resistant to the swarm dynamic reconfiguration when the peers join or leave the network.

Construction of UAVs optimal paths (Ali and Zhangang 2021) over undirected connected graphs is suggested in Ali and Zhangang(2021), where the distance, obstacle avoidance, and terrain/radar parameters are taken into account. The time delays for P2P information communication is determined by the state equation of the system. In the UAV swarm balanced clustering method, presented in Brust et al. (2021), the cluster head stands for the entry point. Basic messages sent from the head to its branches trigger them to accept an additional child or discard the current child. UAVs within the swarm periodically scan the surroundings by using a circular transmission range. A solution to the problem of the swarm leader-followers' relationship is given in Ali et al. (2018). The required formation is implemented over the underlying graph. The swarm communication network is implemented as a weighted matrix, where an introduced general equation determines the flight trajectories of the peers. To solve the swarm path planning problem where the surveyed area is modeled as an undirected weighted graph, Monwar et al. (2018) proposed covering all the available inspection points through the union of all trajectories (modeled as a tree to avoid cycles). Initially, the proposed path planning algorithm sorts all UAVs to their available energy in an ascending order. Another approach to solving the swarm path planning problem is given in Shafiq et al. (2015), where the authors developed a bio-inspired strategy for UAVs' routing algorithm over a regular mesh topology with all the edges of the same length. The UAVs are prearranged in different and nonoverlapped colonies with a hierarchy in each colony with an individual leader UAV and its followers. The best path is found based on the combination of Max-Min, colony optimization and Vicsek algorithms.

A solution to the swarm path-planning problem with optimizing multiple objectives simultaneously can be found in Millar et al. (2023). The proposed approach utilizes a multi-objective reinforcement learning algorithm, also Bayesian belief network approach is used to determine risk indicators. The resultant analysis is weighted through the analytic hierarchy process ranking model. Swarm communication capacity maximization approach using mixed-integer nonlinear programming is given in Javed et al. (2023). An outer approximation algorithm has been suggested to mitigate to achieve near-optimal solutions with reduced convergence time and complexity compared with exhaustive search.

Self-organization, also called spontaneous order in the social sciences, is a process where some form of overall order arises from local interactions between parts of an initially disordered system (Bak et al. 1987). The process can be spontaneous when sufficient energy is available, without the need of control by any external agent. It is often triggered by seemingly random fluctuations, amplified by positive feedback. The resulting organization is wholly decentralized over all system components. As such, the organization is typically robust and able to survive or self-repair substantial perturbation.

A narrower, still very closed concept related to self-organization is the phenomenon of self-ordering of systems. Complex dynamic systems are often self-organizing, and depending on the specified leading groups of properties, they are also called self-regulating, self-adjusting, self-learning, or self-algorithmizable systems.

The Abelian sandpile model (Dhar 1990) is the simplest and analytically tractable model of self-organized criticality. The Abelian group structure and the established one-to-one correspondence between the recurrent states of sandpile model and spanning trees of the underlying graph allows an exact calculation of many of its properties. In Ruelle (2021) a detailed overview of the known results about height probabilities and spacial correlation functions of the model is pre-

sented. In parallel, the research also focuses on the rotor-router model (Priezzhev et al. 1996), where a one-to-one correspondence between the defined recurrent states and the graph spanning trees is observed.

Description of our swarm algorithms and models developed based on our obtained results are given below. The distinguishing characteristic of our approach against the existing solutions is that it meets all the classical requirements imposed on self-organizing systems. In contrast, the current implementations address swarm construction and management specifically. Based on the analysis of available solutions, and to best meet the requirements for UAV swarms' construction, we suggest developing and implementing an optimally distributed software–hardware cloud system aimed at operational management of self-organizing UAV swarms with the following characteristics:

- UAVs are loaded with basic schemes of information full exchange (gossip/broadcast models) to enable constructing swarms of logically linked UAVs with required minimum characteristics (minimum time, calls, and channels). Based on our analytically approved theorems and formulas (Hovnanyan et al. 2013*a*, 2014*a*, 2017), multi-agent modeling algorithms have been designed to measure UAV mission performance (displacement, imaging, and information exchange).
- The methods and algorithms for developing optimal fault-tolerant schemes of information full exchange and constructing multi-agent decentralized, self-organizing systems have been adapted. During the information (captured image or images) full exchange between UAVs, probable communication failures (due to external factors such as landscape obstacles, buildings, structures, mountains, etc.) are neutralized using fault-tolerance schemes and algorithms. The results of our research in this area are summarized in Hovnanyan et al. (2013*b*, 2014*b*, 2013*c*). In the event of a single UAV crash, the swarm is dynamically reconfigured with an adequate structure that meets the optimal conditions and ensures mission continuity.
- The UAVs carry out dynamic area imaging during their quasi-random walk (rotor-router model) and perform a complete and reliable exchange of images via an encrypted internal radio communication. According to the Eulerian walk rotor-router model, the UAVs move in quasi-random trajectories in the sub-areas prescribed by individual "road maps". Several of our theoretical results in this area (Poghosyan and Priezzhev 2014; Papoyan et al. 2015, 2016*a*, 2016*b*) are applied in the development of approvable algorithms to bypass static barriers. The developed algorithms loaded into UAVs guide navigation (shootings) along the allocated spatial network's permissible edges and provide collision avoidance. Work distribution among the UAVs is done automatically. In order to increase the quality of the area's complete image and to reduce the swarm overall flight time, the swarm can be replenished with the required number of UAVs.
- The embedded confidential computing algorithms ensure the swarm data and communication security, strong identification and authentication of the UAVs within the swarm during the flight.
- Due to the resource-limited nature of UAVs, uniform distribution of the computing power is achieved through gossiping algorithms allowing communication solely between the logically linked UAVs.

In our approach, the UAV actions within the swarm are formed according to the network graph downloaded from the virtual cloud server. The nodes on the graph have one arrow each. The swarm UAVs all perform the same operation: transition from a node to a current unvisited neighbor node. The UAV reaching a node rotates that node arrow towards the next neighboring node and moves there. The flight trajectories are formed dynamically during the swarm operation and depend on the actual number of UAVs within the swarm (once replenishment or removal is done), and do not depend on the UAVs' flight speed and the fact that the status of the UAVs may change over time to the striking mode, and, at the end of the mission, the UAVs may rejoin the swarm and continue to follow the current trajectory. Together, the UAV trajectories form an Eulerian circuit, while shooting along all the graph's directed edges is performed exactly once. It is worthwhile to note that gossip schemes are implemented according to the principle of rotors. Each UAV calls a logically linked current neighbor that has not been called/visited yet without authenticating that neighbor. During the calls, only the sender's membership is authenticated. Moreover, in the case of k-tolerance failures, the calls are cyclically repeated k times.

Considering the operational management of swarms in hostile or dangerous environments, the UAVs within the swarm may change the predefined trajectories, update and exchange session keys, etc., or detect hostile nonmembers. Among other cybersecurity concerns, the latter is essential as outsiders may bypass the swarm network and, once impersonated, may try to destroy the swarm cooperative flight.

To solve this problem, we developed algorithms for verifying the swarm membership during the flight. For this purpose, mechanisms for identification and strong authentication of individual UAVs have been designed based on lightweight, secure multi-party computations (MPC) (Maurer 2002; Damgard and Ishai 2005;Canetti 2023). AES-256 is embedded within the UAV hardware to encrypt the captured images, shared secret tokens, session keys, and authentication data. Membership verification is performed using the swarm encrypted blockchain (Wang et al. 2020), where colmputationally heavy proof of work is replaced by an MPC on the embedded Knödel Graph isomorphism. At the fabrication and registration stage, each UAV is assigned a unique identifier registered within the blockchain, a tamper-proof storage ensuring the swarm UAVs reliable distributed authentication. With every UAV replenishment or removal, the blockchain gets updated.

While the blockchain guarantees the tamper-proof of on-chain data, the encrypted state of the blockchain makes it useless for outsiders. The encrypted identity information stored in the blockchain is verified for the swarm members. The UAVs exchange tokens confidentially computed on the

individual hardware during the flight. The UAVs do not exchange the original secrets, which may leak significant information, but exchange and verify the MPC encrypted results over the original secret. These computations' results are distinct as they are performed over UAVs' unique identifiers, which are computationally concealed. This prevents outsiders from mimicking the exchanged data. Also, outsiders cannot generate legitimate tokens as they are unaware of the embedded graph, predefined isomorphism, and embedded confidential computing.

3. Mathematical models underlying the construction of self-organizing swarms of UAVs

To design self-organizing swarms of UAVs for optimal targeting task performance, we use a rotor-router model introduced in Priezzhev et al. (1996) and relevant mathematical preliminaries to facilitate full information exchange (Cooper et al. 2010).

Consider a directed graph (digraph) $G = (V, E)$ with a set of vertices $V = V(G)$ and a set of directed edges $E = E(G)$. We assume that there are no self-loops (edges that connect a vertex to itself) or multiple edges (two or more edges with both the same tail vertex and the same head vertex) in G. If for each edge directed from v to w, there is also a corresponding edge directed from w to v, we call the graph G bidirected. In other words, every edge in a bidirected graph has its opposite edge. A bidirected graph can be obtained from undirected graph by replacing each of its edges with two directed edges, one in each direction. A spanning subgraph G' of a bidirected graph G is a digraph with the set of vertices $V(G') = V(G)$ and a set of edges $E(G') \subseteq E(G)$.

A path of length n from vertex $a \in V$ to $b \in V$ is a sequence of distinct vertices $v_1, v_2, \ldots, v_{n+1}$ such that v_i and v_{i+1} are connected by an edge $e_i \in E$, $i = 1, 2, \ldots, n$, $v_1 = a$, $v_{n+1} = b$. The path becomes a cycle if $a = b$. A shortest possible cycle has length 2 and consists of two adjacent vertices v_1, v_2, which are connected by a pair of edges from v_1 to v_2 and back. We call such cycles dimers by analogy with lattice dimers covering two neighboring vertices. A cycle formed by more than two edges is called contour.

An Eulerian circuit on a finite digraph is a walk that starts and ends on the same vertex and visits each directed edge exactly once. If such a walk exists, the digraph is called Eulerian. A digraph is strongly connected if for any two distinct vertices v, w there are paths from v to w and from w to v. A strongly connected digraph $G = (V, E)$ is Eulerian if and only if for each vertex $v \in V$ in-degree and out-degree of v are equal. Particularly, one-component bidirected graph is Eulerian.

The rotor-router model is defined as follows. Consider an arbitrary connected digraph $G = (V, E)$. Denote the number of outgoing edges (out-degree) from the vertex $v \in V$ by d_v. The total number of edges of G is $|E| = \sum_{v \in V} d_v$. Each vertex $v \in V$ is associated with a rotor, which is directed along one of the outgoing edges from v. The rotor directions at the vertex v are specified by an integer variable α_v, which takes values from $0 \leq \alpha_v \leq d_v - 1$ for $d_v \geqslant 1$.

The set $\rho = \{\alpha_v | v \in V, 0 \leq \alpha_v \leq d_v - 1\}$ defines the rotor configuration. Starting with an arbitrary rotor configuration, one drops a chip to a vertex of G chosen at random. At each time step the chip arriving at a vertex v, first changes the rotor direction from α_v to $(\alpha_v + 1) \mod d_v$, and then moves one step along the new rotor direction from v to the corresponding neighboring vertex.

Figure 1 illustrates three steps of the rotor walk on a square lattice.

The rotor configuration ρ can be considered as a spanning subgraph of G ($\rho \subset G$) with the set of vertices $V(\rho) = V(G)$ and the set of directed edges $E(\rho) \subset E(G)$ coinciding with the rotors. The state of the system at any moment of time is given by the pair (ρ, v) of the rotor configuration ρ and the position of the chip $v \in V$. A vertex $v \in V$ is called a sink if its out-degree $d_v = 0$. In the absence of sinks, i.e., when each vertex has at least one outgoing edge, the motion of the chip does not stop.

If iterating the rotor-router operation from the state (ρ, v) eventually leads back to (ρ, v) we say that (ρ, v) is recurrent; transient otherwise. According to Priezzhev et al. (1996), the rotor-router walk started from an arbitrary initial state (ρ, v) on a finite Eulerian graph, passes transient states and enters into a recurrent state continuing the motion in the limiting cycle, which is an Eulerian circuit of the graph. A rigorous and self-contained survey of the rotor-router model is given in Holroyd et al. (2008).

A connected spanning subgraph of a digraph G, in which every vertex has one outgoing edge contains exactly one cycle. The state (ρ, v) is called unicycle if the set of edges $E(\rho)$ contains a unique directed cycle and v lies on this cycle (Holroyd et al. 2008; Poghosyan and Priezzhev 2014). Then, two basic properties of the rotor-router model on the Eulerian graphs can be formulated in terms of unicycles.

Theorem 1. (Holroyd et al. 2008). Let G be a strongly connected digraph. Then a single-chip-rotor state (ρ, v) on G is recurrent if and only if it is a unicycle.

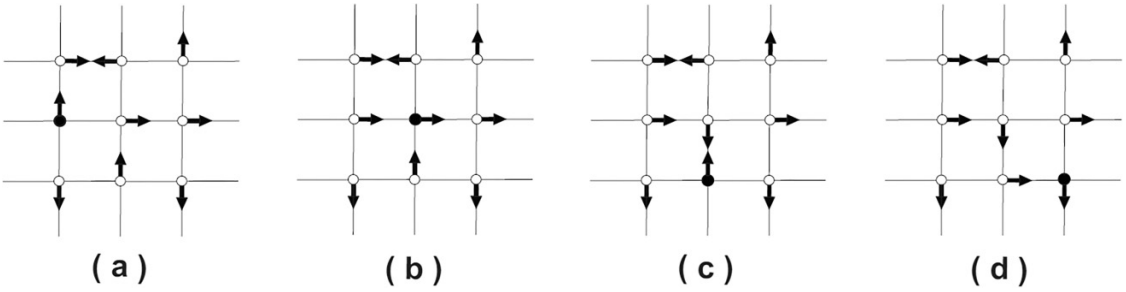
The rotor states that are not unicycles are transient. In contrast to recurrent states, they appear at the initial stage of evolution up to the moment when the system enters into the Eulerian circuit.

Theorem 2. (Holroyd et al. 2008). Let G be an Eulerian digraph with m edges. Let (ρ, v) be a unicycle in G. If one iterates the rotor-router operation m times starting from (ρ, v), the chip traverses an Eulerian circuit of G, each rotor makes one full turn, and the state of the system returns to (ρ, v).

A theorem on reversibility of loops at the recurrent state reads:

Theorem 3. (Holroyd et al. 2008; Poghosyan and Priezzhev 2014). Let G be a bidirected planar graph with the outgoing edges at each vertex ordered clockwise. Let (ρ, v) be a unicycle on G with the cycle C oriented clockwise. After the rotor-router walk makes some number of steps, each rotor internal to C has performed a full rotation, each rotor external to C

Fig. 1. Circles denote the lattice sites. (*a*) The particle is originally in the filled circle where the arrow is directed "up". (*b*) The chip rotates the arrow clockwise and moves right. (*c*) The next clockwise rotation sends the chip down. (*d*) The last position of the chip is in the lower right corner.



has not moved, and each rotor on C has performed a partial rotation so that C is now oriented anti-clockwise.

Consider a bidirected contour $C = (v_1, v_2, \ldots, v_n)$ in a digraph $G = (V, E)$, that is a contour in which the vertices v_i and v_{i+1} are connected by two edges, one in each direction, $1 \le i \le n$, where the n-periodicity is assumed, i.e., $v_{i \pm n} \equiv v_i$. Let e_i^+ and e_{i+1}^- are directed edges connecting v_i to v_{i+1} and v_{i+1} to v_i, respectively. Here, the n-periodicity is also assumed, i.e., $e_{i\pm n}^+ \equiv e_i^+$ and $e_{i\pm n}^- \equiv e_i^-$. The superscripts at e_i^- and e_i^+ are introduced to denote the negative and positive directions at v_i with respect to the contour C, respectively. Given the rotor-router model defined on G, we say that the bidirected contour C obeys the domino ordering if for each rotor at v_i, $1 \le i \le n$ there exists a direction $\alpha_{v_i}^\star$ such that the rotor $\alpha_{v_i}^\star$ points from v_i to v_{i-1} and $\alpha_{v_i}^\star + 1$ points from v_i to v_{i+1}. The directions $\alpha_{v_1}^\star, \ldots, \alpha_{v_n}^\star$ are called negative with respect to C, whereas the directions $\alpha_{v_1}^\star + 1, \ldots, \alpha_{v_n}^\star + 1$ are called positive, respectively.

A weak reversibility is introduced in Papoyan et al. (2015) for rotor-router walks, which start from transient state. In contrast to the case with recurrent initial state, here, after the initial contour is reversed, some rotors inside the contour may not perform a full rotation.

Theorem 4. (Papoyan et al. 2015; Papoyan et al. 2016a). Given an arbitrary finite Eulerian digraph G, let $C = (v_1, \ldots, v_n)$ be a bidirected contour obeying domino ordering. Let the rotor-router walk starts at the vertex v_n from an initial rotor configuration with positive directions of all rotors at v_i ($i = 1, 2, \ldots, n$). Then, after some number of steps, the walk produces a configuration with negative directions $\alpha_{v_1}^\star, \cdots, \alpha_{v_n}^\star$. The moments t_i ($i = 1, 2, \ldots, n$), when the directions $\alpha_{v_i}^\star$ are reached, are ordered as follows: $0 < t_n < t_{n-1} < \cdots < t_2 < t_1 \le |E|$.

However, two reversibility properties are still retained: the walk enters and leaves the contour from the same vertex, and the orientation of rotors on the contour is reversed. If we use the Abelian property of rotor-router walk (Priezzhev et al. 1996), we can generalize the weak and strong reversibility properties for multi-particle walk:

Theorem 5. Given an arbitrary finite Eulerian digraph G, let $C = (v_1, \ldots, v_n)$ be a bidirected contour obeying domino ordering. Let $k < n$ rotor-router particles start their motion at contour vertices $v_1, \ldots, v_n$, so that there is no more than one particle at one vertex. Denote these vertices as $v_{S_1}, \ldots, v_{S_k}$. Assume that the rotors v_i, ($i = 1, 2, \ldots, n$) at the contour C initially have positive directions. During the multi-particle walk, if a particle arrives at one of the vertices v_{S_j}, ($j = 1, 2, \ldots, k$) with negative directed rotor at that vertex, that particle stops its motion. Then, after some number of steps, regardless of the order in which the particles move, the walk produces a configuration with negative directions $\alpha_{v_1}^*, \ldots, \alpha_{v_n}^*$. Moreover, all vertices $v_{S_1}, \ldots, v_{S_k}$ will be occupied by one particle each.

Assuming the initial rotor-router configuration is recurrent, namely, there are no other cycles except C, each rotor internal to C will perform a full rotation at the end of the walk.

Proof. The configuration, in which all particles are located at contour vertices with negative directions can be interpreted as a stable state, since the next step will cause particles to leave the system. The Abelian property of rotor-router walk for the graph G is as follows: Let $\tau_0, \tau_1, \ldots, \tau_n$ is a sequence of rotor-router configurations of G, each of which is a successor of the one before, so that τ_n is stable. If $\tau_0, \tau_1', \ldots, \tau_m'$ is another such sequence, then $m \le n$. If in addition τ_m' is stable, then $m = n$ and $\tau_n = \tau_n'$, and for each vertex w, the number of times the rotor at w changes its direction is the same for both histories.

Let us create two sequences of configurations. In the first sequence, allow only the first particle to move until it reaches a stable state, specifically, the initial vertex with a negative direction of the rotor at that vertex. According to Theorem 4, all vertices on the contour will have negative directions at the end of the motion. In the second sequence, let us consider an arbitrary sequence of particle jumps, which stops when the system reaches a stable state. From the Abelian property, the final states of both sequences are the same.

Algorithms for UAV motions are developed based on the aforementioned theorems, the interpretation of which is given below. The image of the surveyed area is displayed on the user's screen, then a finite square lattice with closed boundary conditions is allocated over that image. For simplicity, it is assumed that the outgoing edges at each vertex are ordered clockwise. The rotors at vertices of the lattice are oriented to form a unicycle with clockwise contour along the

boundary edges. Notice that this contour obeys domino ordering. According to the Theorem 2, if a chip UAV starts its motion from any boundary vertex, it will move in an Eulerian circuit. According to the Theorem 3, the UAV will pass through all the vertices and will perform full rotation of their rotors. At the end of the motion, the UAV returns to the starting point, and the contour turns counterclockwise.

Now let random orientations for the rotors at internal vertices be chosen, which is a transient state of the system (Theorem 1). After quite long UAV motion, a recurrent state is formed (i.e., unicycle), and at this moment all internal cycles are eliminated, according to the Theorem 1. During this motion, the outer contour changes its orientation from clockwise to anticlockwise and vice versa several times (Theorem 4). The iterations of reversing the outer contour stop after the UAV has visited all the vertices and returns to initial vertex. The graph becomes unicycle, so the subsequent trajectory of the UAV motion forms an Eulerian circuit (Theorem 2 and Theorem 3). Once the internal cycles are eliminated, the user places the required number of UAVs on the boundary vertices, i.e., the contour of the unicycle, so that each vertex receives at most one UAV.

According to the Theorem 5, the UAVs together will do the same work as a single UAV would do, if they stop their motion at the boundary vertices with rotors oriented anticlockwise. Therefore, each of them will traverse a part of an Eulerian circuit, and together they will form a full Eulerian circuit. It should be noted that, since we have the Abelian property, the motion speeds of the UAVs are not important: some UAVs may stop the motion later while the others – sooner. In any case, the UAVs will return to the starting points, not necessarily to their own which is the case for a single UAV (Theorem 5).

This motion behavior is programmed and uploaded into the UAVs' memory. All the UAVs operate according to the same loaded graph map. At every time step the UAV rotates the rotors of all the other UAVs in its map, it makes the others move and moves itself with them. Depending on their speed and individual map state, the UAVs may actually be in completely different vertices compared to their own map. In any case, the Eulerian circuit is preserved.

During the motion, if after each time step, the UAVs inform each other whether they have managed to reach the target vertex or not, then the trajectories of the UAVs can be changed. However, the group's mission will be accomplished and the UAVs' trajectories together will form an Eulerian circuit. In the event of a UAV crash, the latter's task is transferred to the first UAV that first completed its task, so it moves to the vertex of the crashed UAV and continues performing its task.

Aimed at uniform distribution of the tasks, a UAV that has completed its task may start from a boundary vertex that was not visited yet. It is equivalent to assuming that an initially crashed UAV was located on that boundary vertex. Similarly, during the operation, the swarm can be replenished with UAVs that will be launched from unvisited boundary vertices.

The number of simultaneous UAV encounters at each vertex is equal to the number of edges adjacent to those vertices. This makes it easier to solve the collision avoidance problem by considering each vertex as a stopover area for four UAVs, or by flying individual UAVs at different heights, or with curvilinear trajectories from vertex to vertex. Other technical solutions may also be introduced.

It is worth to note that this UAV swarm motion scheme is valid and applicable also for arbitrary connected planar bidirected graphs instead of square lattices. The vertices in the graph, which correspond to the obstacles observed during low-altitude flights on the image of the surveyed area, can be removed along with their incident edges, meanwhile preserving the connectivity of the graph. If necessary, new vertices and new edges can be added while maintaining planarity of the graph. After such transformations, the reversibility theorems (Theorem 3, 4, and 5) remain valid.

Development of decentralized and self-organizing swarms of logically linked UAVs involves the design of optimal and fault-tolerant schemes (gossip/broadcast models) enabling dynamic snapshotting and full exchange of captured images of surveilled areas during the swarm quasi-random walk (rotor-router model). The construction is given below regarding essential definitions, concepts, and mathematical models (Poghosyan et al. 2023).

The gossip problem posed decades ago is formulated as follows: each of the n participants within the group possesses distinct information. The goal is to distribute all n messages among all n participants via phone calls. During each phone call, two participants exchange their current messages. The minimum number of required calls is a well-known: $\tau = 2n - 4$, $n > 4$. The problem can be presented in the form of a weighted graph, the vertices of which correspond to the participants, and the weights of the edges indicate the moment of time when the communication among peers took place. Compared to existing solutions, in our models, the communication between any two vertices (peers) occurs instantaneously and requires a single time step. However, these results raised other related issues, particularly, concerning information full exchange among peers with minimum number of channels, minimum number of calls, minimum time, and simultaneous minimization of these parameters.

The involvement of k-fault tolerant gossip graphs allowed us to generalize the gossip problem for at most k arbitrary failures of calls. Note that in case of call failure, no information exchange takes place. The next problem was to determine the minimum number of calls that would ensure k-fault-tolerance between n participants, $\tau(n, k)$. This problem is still open. Up to date, only upper or lower bounds exist for $\tau(n, k)$:

$$\tau(n,k) \le \frac{n}{2}\log_2 n + \frac{nk}{2} \tag{1}$$

for n being a power of 2, and $\tau(n,k) \le 2n\lfloor \log_2 n \rfloor + n\left\lceil \frac{k-1}{2} \right\rceil$, otherwise.

A gossip scheme (a sequence of calls between n nodes) can be represented by an undirected edge-labeled graph $G = (V, E)$, $|V| = n$. The vertices V and edges E of the graph G correspond to nodes and calls between nodes in the gossip scheme, correspondingly. Such graphs may have multiple edges, but no

loops. Edge numbering (labels) of G is a mapping t_G: $E(G) \rightarrow Z^+$, where the weight $t_G(e)$ for an $e \in E(G)$ represents the moment of time when the corresponding call took place.

Definition 1. Two weighted graphs, G and H, are called isomorphic, if there exists a one-to-one correspondent mapping between the set of their nodes, f: $V(G) \rightarrow V(H)$, such that any two vertices u and v are adjacent in the $\{v, w\} \in G \Leftrightarrow \{f(v), f(w)\} \in H$, and $t_G(u, v) = t_H(f(u), f(v))$.

Definition 2. Operation of Local Interchange. Let $E_v(G)$ denotes the set of edges incident to the given vertex v. Given an edge e and a vertex v incident with e, the following two subsets of the $E_v(G)$ are constructed, as follows:

$$P_v^+(e, G) = \left\{ e' \in E_v(G) \,|t_G \quad e' \quad > t_G(e) \right. \tag{2}$$

$$P_v^-(e, G) = \left\{ e' \in E_v(G) \,|t_G \quad e' \quad < t_G(e) \right. \tag{3}$$

The sets of edges $P_v^-(e, G)$ and $P_v^+(e, G)$ correspond to the calls made by node v before and after the moment $t_G(e)$ respectively.

Definition 3. The "permute greater labeled edges operation" $P^+(e)$ on a selected edge $e' \in E_v(G)$ connecting vertices u and v is called the modification of G, resulting in permutation of edges incident to e as follows:

$$E_u\left(P^+(e)\,G \;= P_u^-(e, G) \cup P_v^+(e, G)\right. \tag{4}$$

$$E_v\left(P^+(e)\,G \;= P_v^-(e, G) \cup P_u^+(e, G)\right. \tag{5}$$

Definition 4. The "permute lesser labeled edges operation" $P^-(e)$, on an edge e connecting vertices u and v is defined accordingly, as follows:

$$E_u\left(P^-(e)\,G \;= P_u^+(e, G) \cup P_v^-(e, G)\right. \tag{6}$$

$$E_v\left(P^-(e)\,G \;= P_v^+(e, G) \cup P_u^-(e, G)\right. \tag{7}$$

The operators P^- and P^+ are called "local interchange operators".

The following theorem is known from gossip graphs theory, which plays crucial role in construction of minimal gossip schemes:

Theorem 6. (Hovnanyan et al. 2013*a*, 2014*a*; Hovnanyan 2018) The result of the application of the operators of local interchange on a complete gossip graph is also a complete gossip graph.

There is an open question formulated in our papers (Hovnanyan et al. 2013*a*, 2014*a*; Hovnanyan 2018), which aims to generalize the above theorem to k-tollerant graphs: What are the sufficient and necessary conditions imposed on the graph structure for applying local interchange operators on a k-tolerant gossip graph so as not to affect the fault tolerance level of the graph? We also used the local interchange method for constructing fault-tolerant gossip schemes based on Knödel graphs.

Definition 5. A Knödel graph with $n \geqslant 2$ vertices (n is even) and $1 \leq \Delta \leq \lfloor \log_2 n \rfloor$ degrees is denoted by $W_{\Delta, n}$, where vertices are pairs of type (i, j), $i = 1, 2$; $0 \leq j \leq \frac{n}{2} - 1$. For each of j and l, $0 \leq j \leq \frac{n}{2} - 1$, $l = 1, .., \Delta$, there exists an edge weighted l between $(1, j)$ and $\left(2, j + 2l - 1 - 1 \bmod \frac{n}{2}\right)$ nodes.

Although justified partially in Hovnanyan (2018), it allowed us to introduce hypotheses on the estimate of the minimum number of calls and time steps for fault-tolerant gossip schemes. Benefitting from the symmetry of Knödel graph, these hypotheses have been verified experimentally for various (even quite large) number of vertices. Experimental verifications are carried out using our developed software package, the Graph Plotter (Hovnanyan et al. 2013*c*). The Graph Plotter aims to simulate any "gossip" scheme and obtain ad hoc topologies that will optimally meet the requirements, as well as to experimentally confirm their fully gossiping nature and the level of fault-tolerance. In order to obtain k-fault-tolerant gossip scheme, cyclic iteration of additional k time steps are proposed. An example of constructing a 1-fault tolerant gossip graph is given in the Fig. 2.

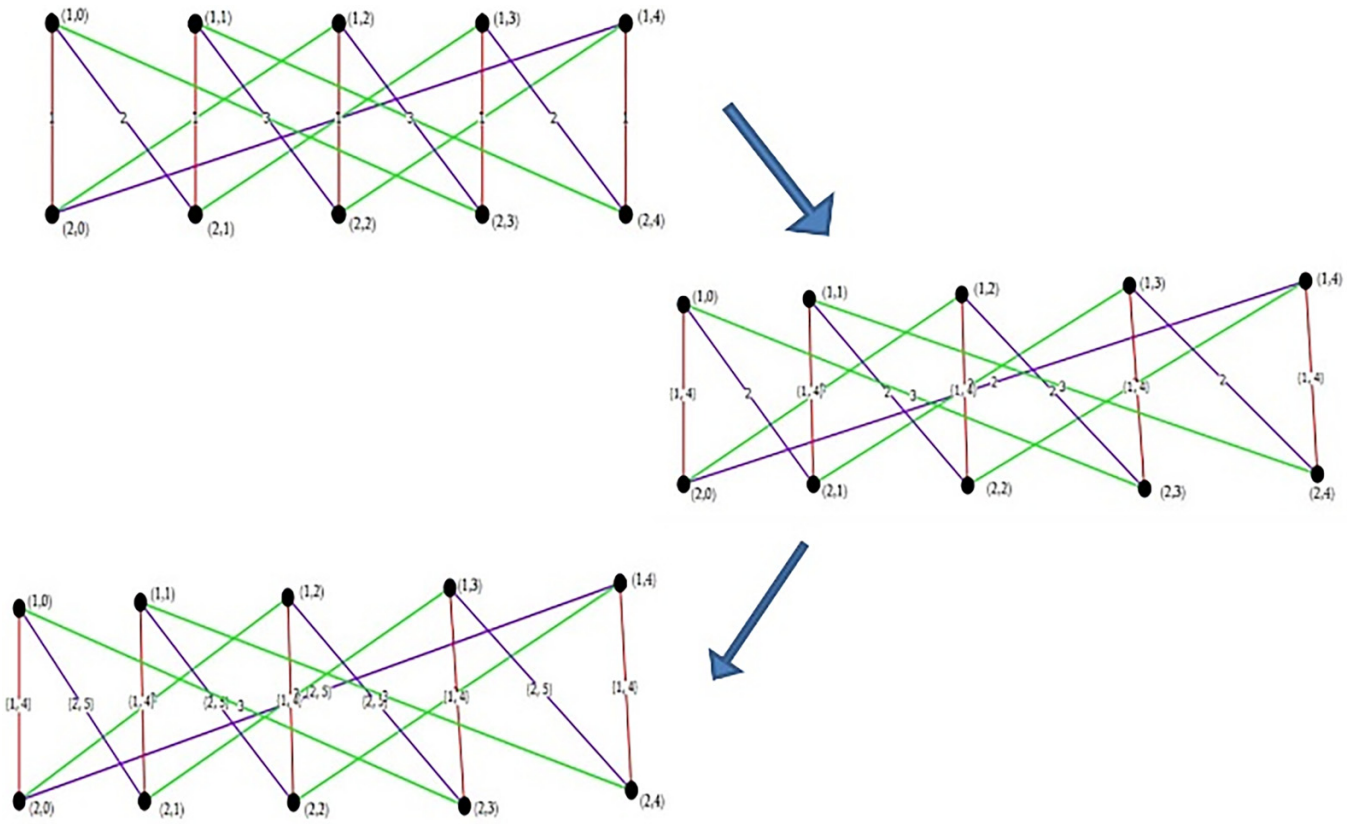
Additionally, the construction described in this section holds $\lceil \log_2 n \rceil + k$ upper bound on the minimum possible time of complete fault-tolerant gossiping. Considering the fact that this result was shown to be a lower bound as well (see Haddad et al. (1987) and Gargano (1992)), we can claim (as a corollary from the above hypothesis) that the minimum time required to complete k-fault-tolerant gossiping is as follows:

$$T(n, k) = \lceil \log_2 n \rceil + k, \text{ if } n \text{ is even} \tag{8}$$

A brief interpretation of the algorithms developed for UAV swarms based on the above results is given below. During the walk of logically connected peers on the gossip graph, the exchange of the following information is carried out: authentication keys, encrypted captured images, and notification about successful completion of transition from one node to another. In situations where connections are broken (resulting in unsuccessful message reception), the Knödel graph is automatically generated with new dimensions. Additionally, there is an increase in the number of time steps for information exchange. Communications may be disrupted not only by obstacles, but also when the two peers are in a distance exceeding the capacity of the radio channel. The situation is addressed by applying the interchange P^+ operator, which will order a new connection. Obviously, provision of a higher fault-tolerance implies performing a larger number of iterations for exchanging information.

The above constructions enable the development of derived models for NOHO gossip graphs, minimal gossip graphs, fault-tolerant gossip graphs, and fault-tolerant gossip graphs over Knödel graphs.

Fig. 2. Iterative construction of the *k*-fault-tolerant graph.



Definition 6. An edge-permuted Knödel graph with $n \geqslant 2$ vertices (n is even) and $1 \leq \Delta \leq \lfloor \log_2 n \rfloor$ degrees is denoted by $M_{\Delta,n}(p)$, where vertices are pairs (i, j) with $i = 1, 2$ and $0 \leq j \leq \frac{n}{2} - 1$. For each of the j and $l = 1, 2, \ldots, \Delta$, there exists l-weighted edge between $(1, j)$ and $\left(2, \left(j + 2^{p+l-1} - 1\right) \bmod (n/2)\right)$ vertices, where $p = 0, \ldots, \Delta - 1$ is an integer parameter of the graph.

Theorem 7. Hovnanyan (2018) The graph

$$G = M_{\lfloor \log_2 n \rfloor, n}(p) + M_{1,n}(p) \tag{9}$$

is a complete gossip graph for any $n \neq 2^k$ and $p = 0, \ldots, \Delta - 1$. This graph is not isomorphic to the graph $W_{\lfloor \log_2 n \rfloor, n} + W_{1,n}$ except for $n = 2^k - 2$ even though both are gossip graphs.

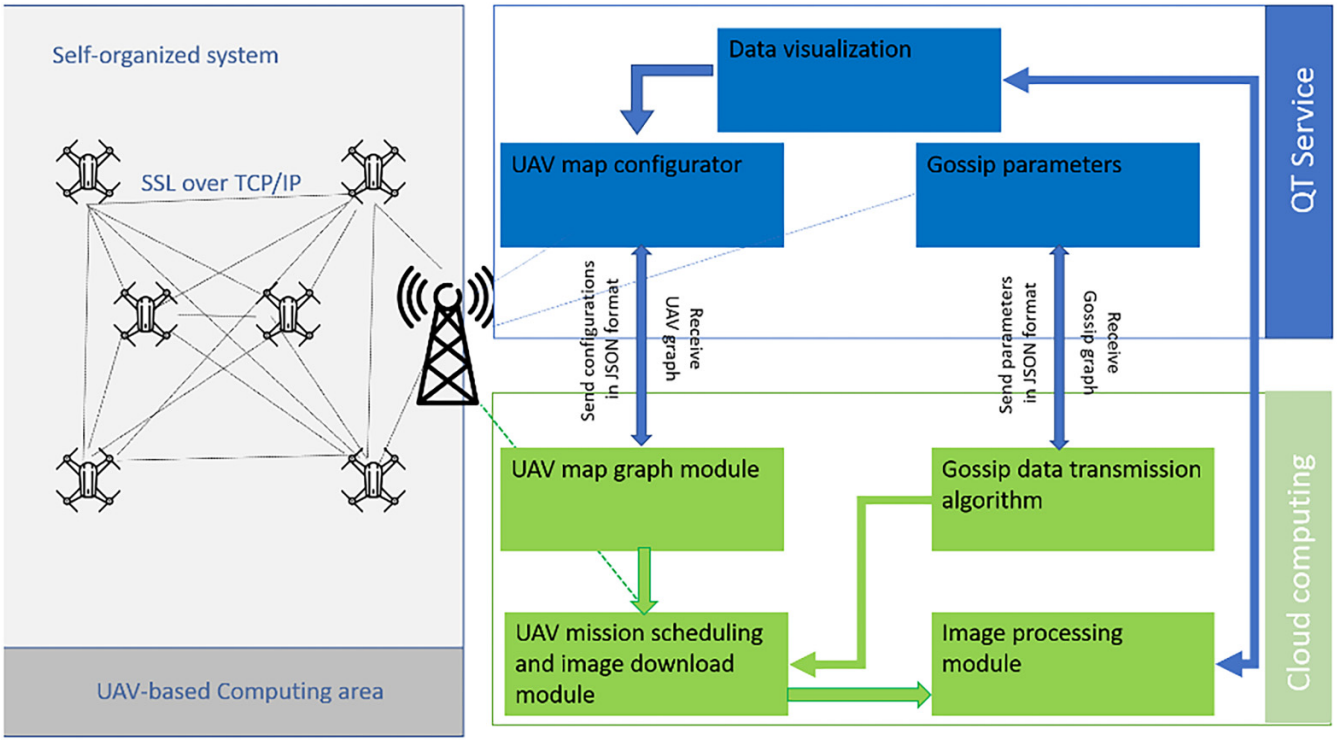
The above information dissemination and full exchange models provide the required level of fault tolerance. Taking into account the resource-constraint nature of the swarm, lightweight cryptographic algorithms, and protocols have been embedded, which ensure encryption of captured data, as well as provision of data integrity during the information full exchange.

4. Cloud platform

The cloud platform consists of a self-organized UAV-based computing area, a cloud computing environment, and QT service layers (refer to Fig. 3). The computing environment utilizes resources from the Armenian research cloud infrastructure (Cloud 2023; Petrosyan and Astsatryan 2022). The Armenian research cloud is built upon the OpenStack and Kubernetes platforms, ensuring robust and scalable performance. OpenStack primarily provides infrastructure as a service capability, managing virtualized resources like computing, storage, and networking. On the other hand, Kubernetes is a container orchestration platform that focuses on automating containerized applications' deployment, scaling, and management. cloud-based mathematical models for self-organizing swarms of UAVs is a customized platform that seamlessly leverages both OpenStack and Kubernetes environments. This tailored approach ensures the utilization of the strengths of both infrastructure paradigms, delivering the requisite flexibility, scalability, and efficiency for our specific applications. The platform encompasses proprietary software codes integrated with third-party tools and libraries. We incorporate OpenDroneMap, a powerful open-source software tool for processing aerial imagery, into our platform. This addition enhances the capabilities of our cloud-based mathematical models, allowing for advanced image processing, 3D reconstruction, and geospatial analysis within the context of UAV swarm operations.

The UAV-based computing area utilizes a network of low-cost Raspberry Pi-equipped UAVs to collect and transfer data to a ground station via Wi-Fi or cellular networks. The collected data received from the UAV cameras is transferred to the ground station in real time. The UAVs communicate through decentralized gossip protocols, eliminating the need for a central control node. Cloud computing has gained trac-

Fig. 3. Cloud platform.



tion in UAV image-processing tasks due to its computational resources and storage capacity. The proposed cloud-based mathematical models for self-organizing swarms of UAVs consist of client and server components.

The QT service layer is responsible for user interaction with the proposed cloud platform. It provides tools and functionalities to visualize, configure, and manage the self-organized UAV swarm. The UAV map configurator is a critical tool for creating and modifying maps for the UAV swarm's navigation and task performance. It enables users to define obstacles, points of interest, and flight paths and optimize the swarm's behavior. The parameter gossip system enables communication and coordination between the UAV swarm, allowing it to operate as a cohesive unit. Additionally, the QT service layer provides visualizations and analytics for users to monitor and understand the UAV swarm's behavior, track its movement, observe its interactions with the environment, and analyze its performance. The QT service layer uses the Internet TCP protocol for secure communication with virtual servers in the cloud infrastructure, utilizing security encryption techniques to protect data transfer from malicious attacks.

The proposed cloud platform incorporates several modules and algorithms to support the self-organized UAV swarm. These modules and algorithms work together seamlessly to provide users with a powerful and efficient computing platform for managing and controlling the UAV swarm.

The UAV map graph module allows users to create and manage maps for the UAV swarm's navigation and task completion. This module reads a JSON file with the client's UAV map specifications, generates a graph representing the map with nodes and edges, and removes any cycles to avoid infinite loops. The Drone Map Processor verifies that the graph meets all the client's specifications, including clear paths between starting and ending positions and safe navigation around obstacles. If verification fails, the module alerts the client and suggests map modifications.

The cloud platform's second module is the UAV mission scheduling and image download module, allowing users to schedule tasks for the swarm and download images for analysis. The module also lets users download images captured by the swarm during their missions, allowing them to analyze and process the data.

The third module is the gossip data transmission algorithm, enabling coordinated communication between the self-organized UAV swarm. This module calculates and selects transmission parameters, creates a node graph, and verifies the validity of the chosen gossip algorithm. The module alerts the system administrator if the graph fails.

The fourth module is the image processing module, providing users with tools for analyzing and interpreting swarm data, including object detection, image classification, and data visualization. An Image Pixel to WGS Converter software program is also used for georeferencing. The proposed cloud platform could also utilize OpenDroneMap as a powerful tool for processing the images captured by the UAV swarm during their missions.

The low-level image processing module recovers images from various degradations, such as noise, blur, haze, etc. Recently, deep neural network-based methods have demonstrated superior performance for image restoration, and the case of multiple degradations can be solved jointly using end-to-end learning. UAVs can also have mounted multispectral

(MS) camera sensors that collect data from various regions of the electromagnetic radiation spectrum with higher spectral resolution than images captured by RGB sensors. When both RGB and MS cameras are in use, one can process data captured by those cameras jointly to generate MS images having high-spatial resolution from high-resolution RGB images and low-spatial resolution MS images. This method, integrated with an out-of-the-box image classification module, provides improved classification accuracy for cases when only one of the cameras (RGB or MS) is in use.

5. Implementation methods and technologies

The methods and technologies underlying UAV swarm design and operational management encompass a series of protocols and algorithms. These mechanisms provide a foundation for modeling swarm intelligence, controlling mission coordination, facilitating collective decision-making, and optimizing targeting task performance.

Distribution and redistribution of the swarm missions are performed automatically. The motion of individual UAVs along edges of the prescribed spatial network is operated by our algorithms, which are developed and embedded accordingly. These algorithms made it possible to avoid near-miss collisions between UAVs within the swarm. Cryptographic algorithms of proven security have been embedded in order to ensure secure communication of UAVs within the swarm.

In case of the lost or crash of individual UAVs, the swarm dynamically reconfigures/reshapes thus ensuring mission continuity. The responsibilities of lost UAVs are automatically transferred to neighbors within the swarm. Appropriate algorithms have been developed to neutralize the individual UAVs localized uncertainties, errors or failures.

Our methods and algorithms for constructing optimal and fault-tolerant schemes implementing information full exchange (gossip/broadcast models) served as a basis for building multi-agent decentralized control systems. During their quasi-random walk (rotor-router model), the UAVs carry out dynamic shooting of the surveyed area. Full and reliable exchange of the captured images is performed via an encrypted internal radio channel. The frequency and other characteristics of the swarm, also the position in space, are determined with the required accuracy, which ensures the surveyed area image completeness and continuity.

In order to resist domain-specific information or physical disruptions throughout the swarm mission performance, the UAV swarm operates without any control panel or external instructions. Nevertheless, appropriate technological solutions have been embedded to establish secure communication with the swarm in order to redirect, guide or return UAVs to the ground station safely. Raspberry Pi 4 Model B types with Robot Operating System are used as on-board computers for achieving autonomous flights. The developed cloud platform integrates application programming interfaces to store, process, and analyze data obtained by UAVs. For this purpose, edge/fog and cloud computing mechanisms are embedded equipped with GPU/TPU cloud-edge devices. The technological solutions are achieved through implementation of broadcast/gossip research problems; Eulerian walk rotor-router models; graphs; cellular automata; coding and cryptography theory along with parallel programming (MPI, OpenMP, CUDA, Chapel) methods.

6. Conclusion

The paper presents an innovative approach to addressing the effective coordination and control challenges of UAV swarms. By introducing mathematical models and paradigms based on self-organizing swarms, the authors have made significant strides in capturing the complex dynamics of UAV swarm behavior. A cloud-based model is proposed to utilize advanced methods and algorithms to develop fault-tolerant schemes and construct a decentralized swarm of UAVs.

Utilizing the rotor-router model for discrete-time walks and incorporating the deterministic evolution of rotor configurations on swarm graph vertices enable optimal imaging of a given area during the swarm's quasi-random walk. The recommended optimal and fault-tolerant gossip/broadcast schemes not only guarantee the information full exchange, but also enhance the swarm's resilience to internal failures and external attacks, while cryptographic protocols ensure security.

The modular nature of our hardware and software solution within the customized virtual environment allows for easy adaptation to various application areas. The model presents a robust and efficient solution for operating UAV swarms in multiple scenarios. It is planned to investigate the scalability of the proposed models and algorithms, and explore methods to optimize the energy consumption of individual UAVs within the swarms.

Future works will be dedicated to extending the rotor-router model aimed at generalization of theorems for non-planar graphs in three-dimensional space and to analytical proof of our hypothesis on minimum number of calls for k-fault tolerant gossip graphs.

To avoid long-distance connections prone to communication failures, we consider it necessary to create a simulation program to be invoked prior to UAVs' flight. By repeatedly changing the numbering of the UAVs placed in their initial positions (generation of the Knödel graph), we model the motion of the UAVs according to the rules for the rotor-router model by iteratively applying the operations of interchange in response to violations of permissible distance of communication. Also, the generation of nonisomorphic graphs preserving the minimum number of calls and time steps should be achieved. Obviously, in this case, there will be fewer repeating time steps and the level of fault-tolerance will decrease.

Acknowledgement

This work was supported by the RA Science Committee, in the frames of the research projects 20TTATRBe016 and 21AG-1B052.

Article information

History dates

Received: 27 June 2023
Accepted: 1 February 2024
Accepted manuscript online: 9 February 2024
Version of record online: 9 April 2024

Copyright

Data Availability

Article does not report data.

Author information

Author ORCIDs

Suren Poghosyan https://orcid.org/0000-0002-4412-4203
Vahagn Poghosyan https://orcid.org/0009-0001-8507-2438
Sergey Abrahamyan https://orcid.org/0000-0002-8099-0205
Artyom Lazyan https://orcid.org/0009-0003-2442-3024
Hrachya Astsatryan https://orcid.org/0000-0001-8872-6620
Yeghisabet Alaverdyan https://orcid.org/0009-0009-1444-8838
Karen Eguiazarian https://orcid.org/0000-0002-8135-1085

Author contributions

Conceptualization: SP, VP, SA, HA, KE
Data curation: HA
Formal analysis: SP, VP, SA, KE
Funding acquisition: VP, HA
Investigation: SP, VP, SA, HA, YA
Methodology: SP, VP, SA, HA, YA, KE
Project administration: SP, VP, HA
Resources: HA
Software: AL
Supervision: SP, VP, HA
Validation: VP, SA, AL, YA, KE
Visualization: AL
Writing – original draft: SP, VP, HA, YA, KE
Writing – review & editing: SP, VP, HA, YA, KE

Competing interests

The authors declare there are no competing interests.

References

https://github.com/LazyanArtyom/DCIS.

Abdenebaoui, L., Kreowski, H.J., and Kuske, S. 2015. Graph-transformational swarms with stationary members. *In* Technological Innovation for Cloud-Based Engineering Systems: 6th IFIP WG 5.5/SOCOLNET Doctoral Conference on Computing, Electrical and Industrial Systems, DoCEIS 2015, Costa de Caparica, Portugal, April 13–15, 2015, Proceedings. Vol. **6**. Springer International Publishing, pp. 137–144. doi:10.1007/978-3-319-16766-4_15.

Ali, Z.A., and Zhangang, H. 2021. Multi-unmanned aerial vehicle swarm formation control using hybrid strategy. Trans. Inst. Meas. Cont. **43**(12): 2689–2701. doi:10.1177/01423312211003807.

Ali, Z.A., Shafiq, M., and Farhi, L. 2018. Formation control of multiple UAVs via decentralized control approach. *In* 2018 5th International Conference on Systems and Informatics (ICSAI). IEEE, pp. 61–64. doi:10.1109/ICSAI.2018.8599350.

Babaoglu, O., Meling, H., and Montresor, A. 2002. Anthill: a framework for the development of agent-based peer-to-peer systems. *In* Proceedings of 22nd International Conference on Distributed Computing Systems. IEEE, pp. 15–22. doi:10.1109/ICDCS.2002.1022238.

Bai, G., Li, Y., Fang, Y., Zhang, Y.A., and Tao, J. 2020. Network approach for resilience evaluation of a UAV swarm by considering communication limits. Reliab. Eng. Syst. Safe. **193**: 106602. doi:10.1016/j.ress.2019.106602.

Bak, P., Tang, C., and Wiesenfeld, K. 1987. Self-organized criticality: an explanation of the 1/*f* noise. Phys. Rev. Lett. **59**(4): 381–384. doi:10.1103/PhysRevLett.59.381.

Bellur, U., and Narendra, N.C. 2006. Towards a programming model and middleware architecture for self-configuring systems. *In* 1st International Conference on Communication Systems Software & Middleware. IEEE, pp. 1–6. doi:10.1109/COMSWA.2006.1665173.

Brust, M.R., Danoy, G., Stolfi, D.H., and Bouvry, P. 2021. Swarm-based counter UAV defense system. Discover Internet of Things, **1**(1): 1–19. doi:10.1007/s43926-021-00002-x.

Campion, M., Ranganathan, P., and Faruque, S. 2018. UAV swarm communication and control architectures: a review. J. Unman. Veh. Syst. **7**(2): 93–106. doi:10.1139/juvs-2018-0009.

Canetti, R. 2023. Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**(1): 143–202. doi:10.1007/s001459910006.

Chen, W., Liu, B., Huang, H., Guo, S., and Zheng, Z. 2019. When UAV swarm meets edge-cloud computing: the QoS perspective. IEEE Network, **33**(2): 36–43. doi:10.1109/MNET.2019.1800222.

Cooper, J., Doerr, B., Friedrich, T., and Spencer, J. 2010. Deterministic random walks on regular trees. Random Struct. Algor. **37**(3): 353–366. doi:10.1002/rsa.20314.

Damgard, I., and Ishai, Y. 2005. Constant-round multiparty computation using a black-box pseudorandom generator. Crypto, **2005**: 378–394. doi:10.1007/11535218.

Dhar, D. 1990. Self-organized critical state of sandpile automaton models. Phys. Rev. Lett. **64**(14): 1613–1616. doi:10.1103/PhysRevLett.64.1613.

Engelbrecht, A.P. 2006. Fundamentals of computational swarm intelligence. John Wiley & Sons, doi:10.5555/1199518.

Gargano, L. 1992. Tighter time bounds for fault tolerant broadcasting and gossiping. Networks, **22**: 469–486. doi:10.1002/net.3230220505.

Haddad, R., Roy, S., and Schaffer, A. 1987. On gossiping with faulty telephone lines. SIAM J. Alg. Disc. Meth. **8**: 439–445. doi:10.1137/0608036.

Holroyd, A.E., Levine, L., Meszaros, K., Peres, Y., Propp, J., and Wilson, D.B. 2008. Chip-firing and rotor-routing on directed graphs. Prog. Probab. **60**: 331–364. doi:10.1007/978-3-7643-8786-0_17.

Hovnanyan, V. 2018. The development of the methods and algorithms for the design of optimal schemes for information dissemination in computer networks. PhD thesis, IIAP NAS Republic of Armenia.

Hovnanyan, V., Poghosyan, S., and Poghosyan, V. 2013a. Method of local interchange to investigate gossip problems. Trans. IIAP NAS RA, Math. Probl. Comput. Sci. **40**: 5–12. Available from http://www.mpcs.sci.am/index.php/mpcs/article/view/243 [accessed 10 December 2021].

Hovnanyan, V., Poghosyan, S., and Poghosyan, V. 2013b. New methods of construction of fault-tolerant gossip graphs. *In* 2013 International Conference on Computer Science and Information Technologies. Revised Selected Papers. IEEE, pp. 1–5. doi:10.1109/CSITechnol.2013.6710341.

Hovnanyan, V., Poghosyan, S., and Poghosyan, V. 2013c. Graph plotter: a software tool for the investigation of fault-tolerant gossip graphs. *In* 2013 International Conference on Computer Science and Information Technologies (CSIT). pp. 20–22.

Hovnanyan, V., Poghosyan, S., and Poghosyan, V. 2014a. Method of local interchange to investigate gossip problems: part 2. Transactions of IIAP NAS RA, Math. Probl. Comput. Sci. **41**: 15–22. Available from http://www.mpcs.sci.am/index.php/mpcs/article/view/225 [accessed 10 December 2021].

Hovnanyan, V., Poghosyan, S., and Poghosyan, V. 2014b. Fault-tolerant gossip graphs based on Wheel graphs. Trans. IIAP NAS RA

Math. Probl. Comput. Sci. **42**: 43–53. Available from http://www.mpcs.sci.am/index.php/mpcs/article/view/214 [accessed 10 December 2021].

Hovnanyan, V., Poghosyan, S., and Poghosyan, V. 2017. Gossiping properties of the edge-permuted Knödel graphs. *In* 2017 International Conference on Computer Science and Information Technologies (CSIT). IEEE, pp. 1–4. doi:10.1109/CSITechnol.2017.8312126.

Javed, F., Khan, H.Z., and Anjum, R. 2023. Communication capacity maximization in drone swarms. Drone Syst. Appl. **11**: 1–12. doi:10.1139/dsa-2023-0002.

Maurer, U. 2002. Secure multy-party computation made simple. SCN, **2002**: 14–28. doi:10.1016/j.dam.2005.03.020.

Millar, R.C., Hashemi, L., Mahmoodi, A., Meyer, R.W., and Laliberte, J. 2023. Integrating unmanned and manned UAVs data network based on combined Bayesian belief network and multi-objective reinforcement learning algorithm. Drone Syst. Appl. **11**: 1–17. doi:10.1139/dsa-2022-0043.

Monwar, M., Semiari, O., and Saad, W. 2018. Optimized path planning for inspection by unmanned aerial vehicles swarm with energy constraints. *In* 2018 IEEE Global Communications Conference (GLOBECOM). IEEE, pp. 1–6. doi:10.1109/GLOCOM.2018.8647342.

Papoyan, Vl.V., Poghosyan, V.S., and Priezzhev, V.B. 2015. A loop reversibility and subdiffusion of the rotor-router walk. J. Phys. A: Math. Theor. **48**(28): 285203. doi:10.1088/1751-8113/48/28/285203.

Papoyan, Vl.V., Poghosyan, V.S., and Priezzhev, V.B. 2016a. Spiral structures in the rotor-router walk. J. Stat. Mech. Theor. Exp. **2016**(4): 043207. doi:10.1088/1742-5468/2016/04/043207.

Papoyan, Vl.V., Poghosyan, V.S., and Priezzhev, V.B. 2016b. Rotor-router walk on a semi-infinite cylinder. J. Stat. Mech. Theor. Exp. **2016**(7): 073209. doi:10.1088/1742-5468/2016/07/073209.

Petrosyan, D., and Astsatryan, H. 2022. Serverless high-performance computing over cloud. Cybern. Inf. Technol. **22**(3): 82–92. doi:10.2478/cait-2022-0029.

Picard, G., and Glize, P. 2006. Model and analysis of local decision based on cooperative self-organization for problem solving. Multiagent Grid Syst. **2**(3): 253–265. doi:10.3233/MGS-2006-2304.

Poghosyan, S., Alaverdyan, Y., Poghosyan, V., Lazyan, A., Hayrapetyan, D., and Shoukourian, Yu. 2023. Algorithms for operating self-organizing swarms of UAVs implementing full exchange of information. AIP Conf. Proc. **2757**(1). doi:10.1063/5.0135806.

Poghosyan, V.S., and Priezzhev, V.B. 2014. Euler tours and unicycles in the rotor-router model. J. Stat. Mech. Theor. Exp. **2014**(6): P06003. doi:10.1088/1742-5468/2014/06/P06003.

Priezzhev, V.B., Dhar, D., Dhar, A., and Krishnamurthy, S. 1996. Eulerian walkers as a model of self-organised criticality. Phys. Rev. Lett. **77**: 5079–5082. doi:10.1103/PhysRevLett.77.5079.

Ruelle, P. 2021. Sandpile models in the large. Front. Phys. **9**: 641966. doi:10.3389/fphy.2021.641966.

Shafiq, M., Ali, Z.A., and Alkhammash, E.H. 2021. A cluster-based hierarchical-approach for the path planning of swarm. Appl. Sci. **11**(15): 6864. doi:10.3390/app11156864.

Sneyd, J., Theraula, G., Bonabeau, E., Deneubourg, J.L., and Franks, N.R. 2001. Self-organization in biological systems. Princeton University Press, doi:10.1515/9780691212920.

Tahir, A., Böling, J., Haghbayan, M.-H., Toivonen, H.T., and Plosila, J. 2019. Swarms of unmanned aerial vehicles — a survey. J. Indus. Inf. Integr. **16**: 100106. doi:10.1016/j.jii.2019.100106.

Wang, H., Li, Y., Zhao, X., and Yang, F. 2020. An algorithm based on Markov chain to improve edge cache hit ratio for blockchain-enabled IoT. China Commun. **17**(9): 66–76. doi:10.23919/JCC.2020.09.006.

Zhou, Y., Rao, B., and Wang, W. 2020. Uav swarm intelligence: recent advances and future trends. IEEE Access, **8**: 183856–183878. doi:10.1109/ACCESS.2020.3028865.

Ziquan, Y., Zhang, Y., Jiang, B., Jun, F.U., and Ying, J.I.N. 2022. A review on fault-tolerant cooperative control of multiple unmanned aerial vehicles. Chin. J. Aeronaut. **35**(1): 1–18. doi:10.1016/j.cja.2021.04.022.