Vasantharaj Rajagopal
Senthil Kumar K ✉

https://doi.org/10.21278/TOF.501075124
ISSN 1333-1124
eISSN 1849-1391

DYNAMIC RECONFIGURATION OF UAV FORMATIONS USING RL-ENHANCED SWARM CONTROL

Summary

Swarms of Unmanned Aerial Vehicles (UAVs) have been widely used in various applications mainly for surveillance and crowd-sensing. A swarm setup is an environment where multiple UAVs coordinate together to execute a specific mission. Such a swarm of UAVs is a useful entity to access areas where human penetration is impossible. To facilitate the need for automation among swarms of UAVs, a Deep Q-Learning based Dynamic Swarm Pattern Formation (DSPF) model is proposed. The proposed Speed Control based Reinforcement Learning (SC-RL) algorithm enhances the DSPF model to achieve pattern formation in an automated manner. The SC-RL algorithm strives to switch patterns efficiently by avoiding inter-UAV collisions and also keeps optimal trajectory intact throughout its pattern switching mechanism. To speed up the pattern building process and enable parallelised coordinate computation, a Decentralised Coordinate Computation (DCC) algorithm is implemented. The Servo Interrupt based Pattern Switch (SIPS) control also gives the DSPF model the ability to change patterns dynamically, which allows it to be adjusted to a variety of situations. By increasing the pattern formation time and distance covered by about 95.68% and 66.67%, respectively, simulations conducted for 100 UAVs demonstrate the viability of the suggested DSPF model in a crowded, collision-prone environment.

Key words: *deep learning; multi-agent system; pattern formation; reinforcement learning; simulation; SWARM*

1. Introduction

Due to their efficiency and accuracy in a variety of commercial and civilian uses, including military conflict, transportation, agriculture, etc., Unmanned Aerial Vehicles, or UAVs, have proven to be a significant benefit across the globe [1]. In 2026, the global UAV market is expected to reach US$58.4 billion, with the Indian market reaching US$1.81 billion. Reactive autonomy [2], Simultaneous Localisation and Mapping (SLAM), and swarming are only a few of the features that UAVs offer and have enormous potential. Among the many features of UAVs, their capacity to form patterns is crucial for their main applications [3]. Because UAV swarms can cooperate, communicate, and share knowledge to improve their chances of survival and combat skills, they are becoming increasingly important in today's flights [4,5]. It is challenging to create effective interactions and conflict strategies when UAV

swarm collaboration and confrontation in adaptive conflict situations are handled by a model that uses multi-agent deep reinforcement learning (MADRL) [6,7]. A swarm of UAVs has the ability to create various patterns during pattern formation, which can be tailored to the many underlying conditions in which they are deployed. For any UAV operation to be carried out and completed effectively, including search and rescue, drone delivery systems, and unmanned surveillance, this dynamic pattern switching is required. Reducing the distance between the drones and the target regions is the operation's main goal [8]. Conventional pattern formation techniques create patterns at predetermined intervals of time by pre-programming the entire mission [9]. These models, however, would not work well in situations where the requirements for patterns are unclear. As a result, the pattern generating system needs to be able to dynamically change patterns as and when needed. Using a holistic reputation model and dynamic intelligent matching, a data-collecting sensing framework for UAVs has been developed that allows for the dynamic tracking of multiple targets. Stackelberg's game theory and equilibrium conditions are used to optimise each aim. Labour economics is used to process the work, while contract theory is employed to plan its future expansion. This approach is widely applicable to crowdsourcing IoT applications and backgrounds in public safety [10]. The mechanism of collision avoidance is one of the most important elements that hinders the creation of an autonomous pattern formation algorithm [11]. A number of collision avoidance systems use dynamic programming techniques, vision-based systems, and information from several sensors installed on the drones [12,13].

Due to the high likelihood of collisions in the pattern generation environment, these systems are ineffective since they need high computing power, which delays the pattern development process. Traditional pattern formation algorithms aim to address this increased processing load by introducing an additional time delay between UAV movements [14]. Unfortunately, these technologies are not suited for use in common multi-drone applications since they sacrifice scalability in order to offer lower processing requirements. Therefore, developing pattern creation techniques that are quick, scalable, and capable of accurately mitigating inter-UAV collisions is crucial.

Furthermore, the Leader-Follower approach is the main focus of a typical swarm pattern creation algorithm. This model is centralised, with the leader drone being the only one to communicate crucial run-time choices to the followers and calculate the positions of the follower drones. This centralised method tends to make the algorithm more time-complex, which lowers the system's overall efficiency.

Given the constraints of the real-time environment, which is made up of complex data with numerous uncertainties, effective self-adaptive methods are necessary for dynamic measurement. Various self-adaptive techniques have been created for particular communication uses. Two scheduling strategies, known as multiple vacation and starting thresholds, have been created to lower the power consumption of Time Division Multiple Access (TDMA), Frequency Division Multiple Access (FDMA), and Code-Division Multiple Access (CDMA) systems to overcome the dependence of random sampling in Next Generation Multiple Access (NGMA) systems. For TDMA and FDMA systems, this technique performs well and is less complicated [15]. It has been suggested that a swarm of unmanned aerial, surface, and underwater vehicles (UAV-USV-UUV) might be used to acquire fine-grained spatial-temporal data for target hunting underwater. To achieve a 95% success rate in energy optimisation, a Deep Q-Learning (DQL) approach is analysed. Nevertheless, this approach must be evaluated in a multi-target and real-time setting [16]. It has been suggested that Gaussian and spherical codebooks be successively refined using an arbitrary memory-less source under the quadratic distortion measure [17]. Under mild moment conditions, a combined excess-distortion probability is attainable. Only the Gaussian codebook is taken into consideration for refining when there are significant variances. Numerous multi-terminal lossy source coding difficulties

can be solved using this technique. In order to convey low latency important traffic in 5G networks, an optimal power allocation technique for non-orthogonal multiple access (NOMA) is suggested. This approach aims to obtain the least amount of power while maintaining dependable transmission. For a small number of receivers, the receiver's Bit Error Rate (BER) is enhanced. For minimal transmission delay and resource optimisation, an Integrated Sensing and Communication (ISAC) network based on low complex graph theory has been created. The suggested approach is useful in providing an alternative means to prevent intolerable router data transfer delays [18].

A brief overview of the current study is outlined below. UAVs have proven to be a significant benefit worldwide. Their ability to operate in swarms, where they can cooperate, communicate, and share knowledge, enhances their chances of survival and improves their combat capabilities, making them increasingly vital in modern warfare. A UAV swarm has the ability to form various patterns during pattern formation, which can be adapted to suit the diverse conditions in which they are deployed. For any UAV operation, such as search and rescue missions, drone delivery systems, or unmanned surveillance, to be effectively carried out and completed, the ability to dynamically switch between patterns is essential. Therefore, the pattern generating system must be capable of adapting in real time as the situation demands. Numerous collision avoidance systems have been developed, incorporating dynamic programming techniques, vision-based approaches, and input from multiple sensors installed on each drone. It is crucial to develop pattern creation techniques that are fast, scalable, and capable of accurately preventing inter-UAV collisions. Given the constraints of real time environments, which are characterised by complex data and numerous uncertainties, effective self-adaptive methods are necessary for dynamic measurement and reliable performance. In addition to aerial applications, it has been proposed that a swarm of unmanned aerial, surface, and underwater vehicles (UAV USV UUV) be utilised to gather fine-grained spatial temporal data for underwater target hunting. For such missions, minimising transmission delay and optimising resources are critical. To this end, an Integrated Sensing and Communication (ISAC) network based on low complexity graph theory has been developed to enhance performance and coordination among the swarm elements.

The objective of the present study is to prevent inter-drone collisions during pattern formation and formation transitions by implementing collision avoidance strategies through dynamic reconfiguration. Additionally, the study explores the use of Deep Q-Learning to enable adaptive and efficient dynamic swarm formation. The goal of the SC-RL model proposed in the present study is to form the required pattern in the shortest possible time without any collision. The structure of the paper is as follows. Section 2 reviews existing methodologies for UAV pattern formation, obstacle detection, and the application of reinforcement learning within UAV swarms. Section 3 provides a comprehensive explanation of the proposed DSPF model, detailing the primary SC-RL algorithm and its associated submodules: the Decentralised Coordinate Calculation (DCC) algorithm and Servo Interrupt based Patten Switching (SIPS) control. Section 4 discusses the simulation results, offering a detailed performance analysis. Lastly, Section 5 concludes the study and outlines potential future research directions.

2. Related work

A swarm of UAVs is a concept that paves the way for coordinated UAV missions [19,20,21] in variable ecosystems. Communication and coordination among drones employed in the system must be established [22]. The swarm reinforcement learning method, inspired by multipoint search optimisation, has shown to be effective in fully observable MDPs. This paper extends it to POMDPs using HQ-learning, demonstrating superior performance through experiments, so that multiple drones will act as a single unit. This communication is generally facilitated with the help of the MAV Link protocol [23]. A critical function of such a system is

the Pattern Formation. Any multi-UAV configuration setup consists of a Base Station (BS) and a Ground Control Station (GCS), which provide grounds for wireless communication and centralised control of the swarm of UAVs The work in [24] proposed the idea of a coordinated multi-UAV pattern formation mechanism that involves consideration of various physical parameters like wind speed, wind direction, obstacles, etc.

Making the multiple UAVs work as a single unit enables pattern formation that leads to the efficient execution of UAV missions [25]. Before pattern formation, the task needs to be grouped into multiple closely related clusters based on its geographical location [26]. During pattern formation, the coordination among UAVs is of prime importance. Leader-follower formation flight [27,28] is an efficient mechanism for inter-UAV coordination. Under this mechanism, a single drone is selected as leader and the rest as followers. The leader broadcasts its positional coordinates and other state manipulation functions to all the followers, based on which critical decisions are taken [29,30]. With an increase in the number of UAVs in the pattern formation environment, it becomes complicated to control the system manually. There is a critical need for automation, which is achieved by Reinforcement Learning (RL) [31,32,33]. In the UAV field, RL provides methodologies for autonomous UAV navigation, collision avoidance, and the stable hovering of drones [34,35,36].

Another determining factor, along with the application of autonomous pattern formation algorithms, is the mechanism of collision avoidance. The collision avoidance mechanism of multiple drones is based on the UAV relative distance calculation model [37,38,39]. Every agent in a multi-UAV system considers only the relative positions of its nearest neighbours both on the left and right. The work in [14] instructs the drones to fly to their destinations at different timings to avoid collision. In the absence of a Global Positioning System (GPS), the collision avoidance mechanism can be executed with the help of a camera and a laser detector [40]. However, these mechanisms pose serious processing demands and exhibit poor scalability, thereby making them inefficient in pattern formation environments. Controlling multiple UAVs with a Micro Air Vehicle Link protocol for a Model-View-Controller (MVC) pattern is proposed in [41]. A new MAV Link message has been established to control a group of UAVs. However, this method needs an improvement in the communication channel. The communication channel often becomes overloaded.

In summary, previous studies span multi-agent deep RL for swarm tactics, dynamic sensing via Stackelberg games, collision-avoidance with vision and dynamic programming, and leader-follower patterning that trades speed for scalability. Energy-aware scheduling, NOMA power allocation, and ISAC graph techniques further highlight the quest for low-latency, power-efficient coordination across 5G-enabled UAV, USV, and UUV networks. Yet none jointly address real-time pattern switching, decentralised computation, and collision mitigation for variable-size swarms. Table 1 shows a comparison of different patterns with their advantages and disadvantages, indicating the need for automatic pattern formation and control for multi-agent systems.

3. Methodology

In a multi-UAV environment, pattern formation is an essential function that helps in making UAV missions efficient and cost effective. The proposed Dynamic Swarm Pattern Formation (DSPF) model helps automate this pattern formation process and ensures that collision is avoided throughout the UAV mission. This automation process would considerably reduce human intervention, saving much manual work such as waypoints, tracking of UAV trajectories for collisions, etc. The DSPF model comprises two functionalities, namely SIPS control and DCC mechanisms. The DSPF is also simulated and evaluated using a Software in the Loop (SITL) setup.

3.1 Software in the Loop (SITL) Setup

In the SITL environment, the proposed DSPF model has been tested over a multitude of scenarios. The SITL environment provides a ground for effectively simulating any swarm of UAV modules, replacing all the hardware components with a corresponding software module. This environment helps to reduce the testing costs and expand the number of scenarios to which the given module can be exposed.

The proposed DSPF model manipulates the physical simulation control layer to achieve efficient pattern formation based on the input data obtained from the GCS via the Ardupilot. Fig. 1 illustrates the various modules in a SITL architecture for the DSPF model. Here, the GCS and telemetry port modules are executed on MAVProxy software that helps to establish overall control of the swarm system and collect valuable data during mission execution. The obtained telemetry data can also be shared with multiple other GCS modules. The Ardupilot module serves to provide a user interface, enabling communication among the user, the GCS, and the physical control module, such as drone physics simulations, Flight Gear, etc. These physical control modules contain built-in functions that help to replicate the UAV's motion and operation. A transmission control protocol (TCP) provides the communication between the Ardupilot, the GCS, and the Telemetry port modules, while the User Datagram Protocol (UDP) is used for the rest of the communications [41][42].

Table 1 Comparison of different pattern switching algorithms.

Ref No.	Algorithm	Advantages	Disadvantages
[4]	Many-to-Many Learning Matching (MILMA), Fast Potential Matching Without Substitutability (HPMA)	HPMA provides a convergence performance robust to dynamic UAV communication networks.	Dynamic situation of drone needs to be analysed.
[5]	Oxyrhynus Marina-Inspired Search (OMS), Circular or Elliptical	Performs both exploration and exploitation of the search space using Levy flight and Brownian search.	The complexity of the formation leads to high computational and communication overhead.
[7]	Artificial Immune System (AIS), Particle Swarm Optimisation (PSO), Virtual Bee Algorithm (VBA)	Solves NP-hard problem. Provides best results compared to classical combinatorial optimisation methods.	Does not consider trajectory optimisation.
[14]	Multiverse Optimiser (MVO)	Path planning is superior to other metaheuristic algorithms.	Produces noise in the communication phase.
[16]	Leader-Follower Consensus Algorithm, 2 Drones	Consensus can be achieved in multivehicle missions under uncertain systems with some conditions.	It may not be suitable for scenarios where the formation needs to adapt to changing conditions or obstacles.
[20]	Sophisticated Path Planning and Collision Avoidance Algorithms, Circle and Fountain Pattern	Application of reinforcement learning deeply analyses the state space, action space, and environment model.	The obtained control decision time is 100 ms.
[25]	Deep Reinforcement Learning (DRL)	It optimises the trajectory of the UAV-RS to minimise usage probability and maintains the network satisfaction index.	Computationally expensive and requires large amounts of data to train the model.
[28]	Agent Learning-Based Clustering Algorithm (ALBCA)	Improves the network stability by facilitating adaptive clustering.	Nodes in the transmission range affect the decision on route selection.

3.2 Servo Interrupt based Pattern Switching (SIPS) control

The outermost layer to the proposed DSPF model for deciding the pattern to be formed at a specific juncture is the interaction between the user and the Multi-UAV system. The traditional and most common approach for deciding patterns is to pre-programme them in a time-interleaved sequential manner. This time-based sequential pattern formation is not a feasible mechanism for all situations where the swarm needs to switch to a desired pattern on the fly. To cater for such scenarios, a SIPS control mechanism is proposed to easily switch to the desired pattern on the fly. This servo or a number of servos can be controlled directly via the mission planner through commands sent from the GCS [42]. The servo input consists of many Radio Frequency (RF) channels, also called the servo source. Each RF channel can be programmed to form a specific pattern under a given scenario.

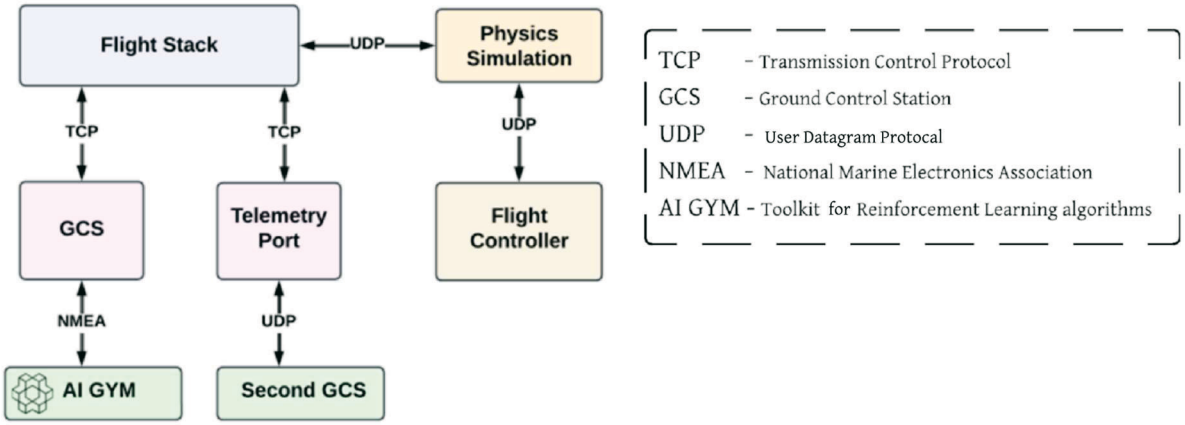

Fig. 1 SITL architecture

Algorithm 1 illustrates the servo-based pattern switching mechanism for a general quadcopter [9] for a sample of three shapes, namely square, triangle, and diamond. In a quadcopter, the first four RF channels are reserved for specific drone motion commands, and hence channels 5, 6, 7, and 8 are used for the pattern formation. The system listens for a servo interrupt, based on which it determines the appropriate servo source value. The leader drone is instructed to switch to a specified formation whenever a toggle of a particular servo source occurs. A toggle in the value of servo source 8 instructs the swarm of drones to return to their initial take-off location, and the drones eventually land. Thus, the SIPS mechanism serves to dynamically form patterns in a multi-UAV environment on the fly.

Algorithm 1 ServoInterrupt Based Pattern Formation

Input: The servo interrupt signal, *k*
Output: Pattern formation based on the servo interrupt

1. **Procedure** *ServoBasedPatternFormation* {k}
2. Let the source of the servo interrupt be *servoSource*
3. **if** *servoSource* == 5 **then**
4. Switch to a square pattern
5. **else if** *servoSource* == 6 **then**
6. Switch to a triangle pattern

```
7.  else if servoSource == 7 then
8.      Switch to a diamond pattern
9.  else If servoSource == 8 then
10.     Return to home and land
```

3.3 Decentralised Coordinate Calculation (DCC)

Once the specified pattern has been input into the system through servo interrupts, it is essential to determine the coordinates of each UAV in the swarm system. The general UAV coordinate calculation algorithm in a leader-follower environment would take place in a centralised manner [9]. In such a scenario, the leader would be responsible for calculating the positions of the followers based on its position and would then communicate the coordinates to the followers. The followers' task would be to receive the calculated coordinates from the leader and then move to the specified destination. In this scenario, the complete overhead lies on the leader, which reduces the efficiency of the system. When the number of drones increases on a large scale, the complexity of the coordinate calculation algorithm increases to O(n).

To avoid such an overhead, the DCC algorithm is proposed. Fig. 2 shows the flow of control of the proposed DCC algorithm. The GCS specifies the pattern to be formed to the leader drone. The leader drone, upon receiving the pattern information, broadcasts the pattern information and its positional coordinates to all the follower drones. Each follower is pre-assigned with its unique ID before the start of the process to avoid conflicts between two drones' positions during the coordinate calculation. The follower uses its unique ID, the leader's positional data, and the pattern formation information to calculate its absolute position. Once the destination coordinates are obtained, the followers move to the specified location independently. When all the followers reach the destination, they issue feedback to the leader. When the leader receives the feedback from all the followers, it ensures pattern formation, and this information is made known to the GCS. Since the followers have a unique ID, the coordinate calculation mechanism for each follower becomes independent, and hence it is parallelised. This makes the DCC algorithm independent of the number of drones, reducing the complexity to O(1).

3.4 DCC for square, diamond and triangle patterns

The DCC algorithm is the central part of the pattern formation mechanism. The algorithm operates in a parallel fashion and calculates the destination positions for all the drones based on the leader's coordinates, the drones' unique IDs, and the information about the pattern to be formed. The implementation of the proposed DCC algorithm has been developed for three shapes, namely square, diamond, and triangle, and has been designed to operate for any number of drones.

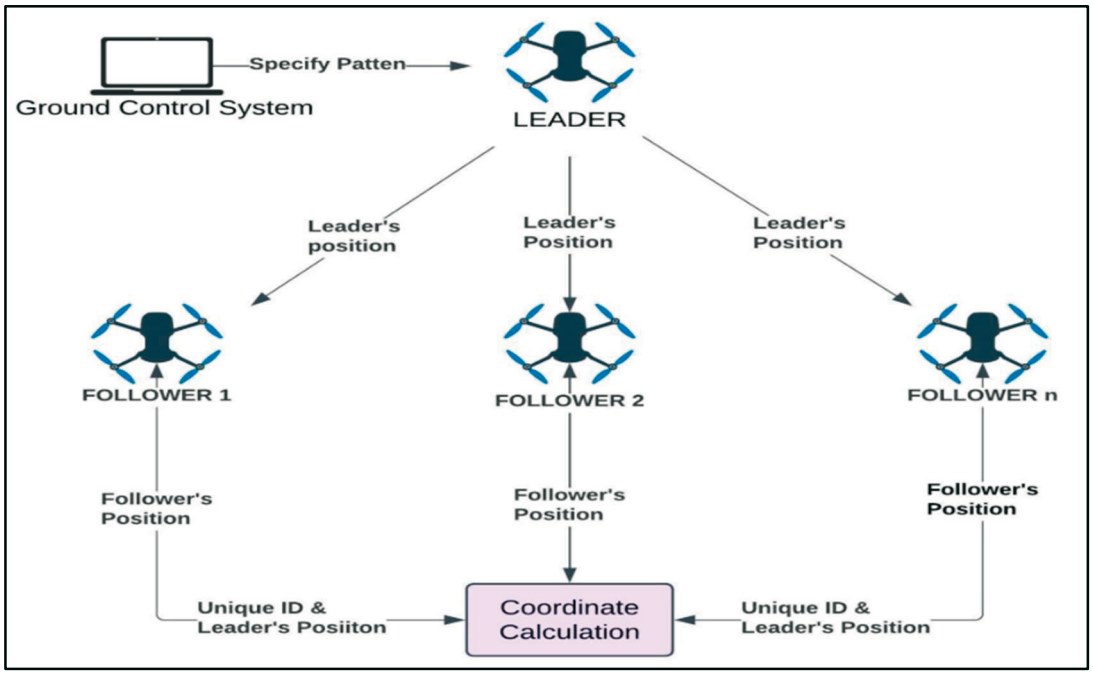

Fig. 2 Distributed coordinate calculation mechanism

Table 2 Notations of the variables used in patterns

S. No.	Variable	Function
1	r	Displacement from the leader
2	θ	Azimuth angle
3	l	Edge length
4	n_{de}	Number of drones per edge
5	n_d	Number of drones
6	n_r	Remaining number of drones
7	k	Coordinate calculation ID

As shown in Fig. 3, the shapes, namely square, diamond, and triangle, are considered to contain four sides, and drones are equally divided among these sides. The edge length 'l' is calculated as the product of the highest number of drones in an edge and the minimum inter-drone distance. The number of drones per edge n_{de} is calculated as (Eq 1):

$$n_{de} = \frac{n_d}{4} \tag{1}$$

With this value of n_{de} and the unique ID, the drone positions are calculated by the followers. In addition, the remaining number of drones nr is calculated as (Eq 2):

$$n_r = n_d - 4 \cdot n_{de} \tag{2}$$

Square pattern

The drones are arranged in a square grid formation, as illustrated in Fig 3(a). Within this configuration, the side length is defined as $l=10*n$ metres, based on a constant inter-drone spacing of 10 metres. For a drone with a unique identifier k, the polar coordinates (r,θ) are determined according to the following cases:

Case I: $n_r=0$ i.e. n_d is an integral multiple of 4.

The following equations represent the positional coordinates of drones when the total number of drones n_d is an integral multiple of 4, i.e., $n_r=0$

$$r = k \cdot 10,\ \theta = 90°,\ \text{where } 1 \le k \le n_{de} \tag{3}$$

$$r = \sqrt{l^2 + (t \cdot 10)^2},\quad \theta = 90° + \tan^{-1}\left(\frac{t \cdot 10}{l}\right) \tag{4}$$

where $n_{de} < k \le 2n_{de}$, $t = k - n_{de}$

$$r = \sqrt{l^2 + ((n_{de} - t) * 10)^2}$$

$$\theta = 90° + \tan^{-1}\left(\frac{l}{(n_{de} - t) * 10}\right) \tag{5}$$

where $2n_{de} < k \le 3n_{de}$, $t = k - 2n_{de}$

$$r = (n_{de} - t) * 10, \theta = 180°$$

where $3n_{de} < k < 4n_{de}$, $t = k - 3n_{de}$ (6)

Eqs. (3), (4), (5), and (6) present the mechanism for calculating the positions of drones when the unique ID, k, is between different ranges, as mentioned.

Case II: $n_r=1$

When the remaining number of drones $n_r=1$, then the additional drone is present in side 2 of the square. The equations for the coordinate calculation for a drone with a unique ID, k, changes as follows:

$$r = k * \left(\frac{l}{n_{de}}\right),\ \theta = 90°,\ \text{where } 1 \le k \le n_{de} \tag{7}$$

$$r = \sqrt{l^2 + (t * 10)^2}, \theta = 90° + \tan^{-1}\left(\frac{t * 10}{l}\right), \tag{8}$$

where $n_{de} < k \le 2n_{de} + 1$, $t = k - n_{de}$.

$$r = \sqrt{l^2 + \left(t * \left(\frac{l}{n_{de}}\right)\right)^2},$$

$$\theta = 90° + \tan^{-1}\left(\frac{n_{de}}{n_{de-t}}\right), \tag{9}$$

where $2n_{de} + 1 < k \le 3n_{de} + 1$, $t = k - (2n_{de} + 1)$

$$r = (n_{de} - t) * \frac{l}{n_{de}},\ \theta = 180°, \tag{10}$$

where $3n_{de} + 1 < k < 4n_{de} + 1$, $t = k - (3n_{de} + 1)$.

In Eqs. (7), (8), (9) and (10), the length of the side of a square, $l=10*(n_{de}+1)$. The ranges of the equations have also been adjusted to accommodate the extra drone's positional calculation.

Case III: $n_r=2$

When the remaining number of drones, $n_r=2$, the additional drone is present in side 4 of the square, in contrast to the previous configuration. Here, Eqs. (7), (8), and (9) remain the same, but Eq. (10) is changed as follows:

$$r=\left(n_{de}-t+1\right)*10, \theta=180°, \quad \text{where } 3n_{de}+1<k\le 4n_{de}+1,\ \ t=k-\left(3n_{de}+1\right) \tag{11}$$

Case IV: $n_r=3$

When the remaining number of drones, $n_r=3$, then the additional drone is present in side 3 of the square, in contrast to the previous configuration. Here, Eqs. (7) and (8) remain the same; however, Eqs. (9) and (11) are modified as follows: Eq. (12) and Eq. (13):

$$r=\sqrt{l^2+\left(n_{de}-t+l^2\right)}$$

$$\theta=90°+\tan^{-1}\left(\frac{l}{\left(n_{de}-t+1\right)*10}\right), \quad \text{where } 2n_{de}+1<k\le 3n_{de}+2,\ \ t=k-\left(2n_{de}+1\right) \tag{12}$$

$$r=\left(n_{de}-t+1\right)*10,\ \theta=180°, \quad \text{where } 3n_{de}+2<k\le 4n_{de}+2,\ \ t=k-\left(3n_{de}+2\right). \tag{13}$$

Thus, in the square formation, the azimuth angle is constant across all drones that are part of side 1 and side 4, having values of 90° and 180°, respectively .

Diamond pattern

The diamond shape as shown in Fig 3(b) is assumed to be a square which has been tilted by 45°. Here, the value is calculated for a drone with a unique ID, k, for the following cases:

Case I: $n_r=0$ i.e. n_d is an integral multiple of 4.

Eqs. (3), (4), (5) and (6), with an addition of 45° to the azimuth angle, θ, determine the drones' positions in a diamond when the total number of drones n_d is an integral multiple of 4, i.e. $n_r=0$.

Case II: $n_r=1$

When the remaining number of drones, $n_r=1$, then the additional drone is present in side 2 of the diamond. The equations for the coordinate calculation of a drone with a unique ID, k, changes as follows:

$$r=k*\left(\frac{l}{n_{de}}\right),\ \theta=135°,\ \text{where } 1\le k\le n_{de} \tag{14}$$

$$r=\sqrt{l^2+(t*10)^2},\ \theta=135°+\tan^{-1}\left(\frac{t*10}{l}\right), \quad \text{where } n_{de}<k\le 2n_{de}+1,\ t=k-n_{de} \tag{15}$$

$$r=\sqrt{l^2+\left(\left(n_{de}-t\right)*\left(\frac{l}{n_{de}}\right)\right)^2}$$
$$\theta=225°-\tan^{-1}\left(\frac{n_{de}-t}{n_{de}}\right) \quad (16)$$
$$\text{where } 2n_{de}+1<k\le 3n_{de}+1,\ t=k-\left(2n_{de}+1\right)$$

$$r=\left(n_{de}-t\right)*\left(\frac{l}{n_{de}}\right),\ \theta=225° \quad (17)$$
$$\text{where } 3n_{de}+1<k<4n_{de}+1,\ t=k-\left(3n_{de}+1\right)$$

In Eqs. (14), (15), (16) and (17), the length of the side of the diamond $l=10*(n_{de}+1)$. This value of l remains the same for the further equations as well.

Case III: $n_r=2$

When the remaining number of drones, $n_r=2$, then the additional drone is present in side 3 of the diamond, in contrast to the previous configuration. Here, Eqs. (14) and (15) remain the same. However, Eqs. (16) and (17) are changed as follows:

$$r=\sqrt{l^2+\left(\left(n_{de}-t+1\right)*10\right)^2},$$
$$\theta=225°-\tan^{-1}\left(\frac{\left(n_{de}-t+1\right)*10}{l}\right) \quad (18)$$
$$\text{where } 2n_{de}+1<k\le 3n_{de}+2,\ t=k-\left(2n_{de}+1\right)$$

$$r=\left(n_{de}-t\right)*\left(\frac{l}{n_{de}}\right),\ \theta=225° \quad (19)$$
$$\text{where } 3n_{de}+2<k<4n_{de}+2,\ t=k-\left(3n_{de}+2\right)$$

Case IV: $n_r=3$

When the remaining number of drones, $n_r=3$, then the additional drone is present in side 4 of the diamond, in contrast to the previous configuration. Here, Eqs. (14), (15) and (18) remain the same, whereas Eq. (19) is modified as follows (Eq. 20):

$$r=\left(n_{de}-t+1\right)*10,\ \theta=225° \quad (20)$$
$$\text{where } 3n_{de}+2<k\le 4n_{de}+2,\ t=k-\left(3n_{de}+2\right)$$

Thus, in the diamond formation, the azimuth angle is constant across all drones that are part of side 1 and side 4, having values of 135° and 225°, respectively.

Triangle pattern

The triangle considered in this formation is a right-angled triangle at vertex 'A'. Even though the triangle is a three-sided figure, we consider it contains four sides, with the hypotenuse being divided into two equal sides as shown in Fig 3(c). The length of half of the hypotenuse is $l=10n_{de}$, due to a gap of 10 metres between each drone. The lengths of each of the other two sides of the triangle are equal to $l\sqrt{2}$. Here, the value of (r,θ) is calculated for a drone with a unique ID, k, for the following cases:

Case I: $n_r=0$ i.e. n_d is an integral multiple of 4.

The following equations represent the positional coordinates of drones when the total number of drones n_d is an integral multiple of, i.e., $n_r=0$.

$$r=k\ *10\sqrt{2},\ \theta=135°,\ \text{where } 1\le k\le n_{de} \tag{21}$$

$$r=\sqrt{l^2+\left(\left(n_{de}-t\right)*10\right)^2}$$
$$\theta=180°-\tan^{-1}\left(\frac{n_{de}-t}{n_{de}}\right) \tag{22}$$
$$\text{where } n_{de}<k\le 2n_{de},\ t=k-n_{de}$$

$$r=\sqrt{l^2+(t\ *10)^2},\ \theta=180°+\tan^{-1}\left(\frac{t}{n_{de}}\right) \tag{23}$$
$$\text{where } 2n_{de}<k\le 3n_{de},\ t=k-2n_{de}$$

$$r=\left(n_{de}-t\right)*10\sqrt{2},\ \theta=225° \tag{24}$$
$$\text{where } 3n_{de}<k<4n_{de},\ t=k-3n_{de}$$

Eqs. (21), (22), (23) and (24) present the mechanism for calculating the positions of drones when the unique ID, k, is between different ranges as mentioned.

Case II: $n_r=1$

When the remaining number of drones, $n_r=1$, then the additional drone is present in side 2 of the triangle. The equations for the coordinate calculation for a drone with a unique ID, k, changes as follows:

$$r=k\sqrt{2}\ *\left(\frac{l}{n_{de}}\right),\ \theta=135°,\ \text{where } 1\le k\le n_{de} \tag{25}$$

$$r=\sqrt{l^2+\left(\left(n_{de}-t+1\right)*10\right)^2},$$
$$\theta=180°-\tan^{-1}\left(\frac{n_{de}-t+1}{n_{de}+1}\right), \tag{26}$$
$$\text{where } n_{de}<k\le 2n_{de}+1,\ t=k-n_{de}$$

$$r=\sqrt{l^2+\left(t\ *\left(\frac{l}{n_{de}}\right)\right)^2}$$
$$\theta=180°+\tan^{-1}\left(\frac{t}{n_{de}}\right), \tag{27}$$
$$\text{where } 2n_{de}+1<k\le 3n_{de}+1,\ t=k-\left(2n_{de}+1\right)$$

$$r=\left(n_{de}-t\right)*\left(\frac{l}{n_{de}}\right)*\sqrt{2},\ \theta=225°$$
$$\text{where } 3n_{de}+1<k<4n_{de}+1,\ t=k-\left(3n_{de}+1\right) \tag{28}$$

In Eqs. (25), (26), (27) and (28), the length of the non-hypotenuse sides of the triangle $l=10*(n_{de}+1)$. This value of l remains the same for the further equations as well.

Case III: $n_r=2$

When the remaining number of drones, $n_r=2$, then the additional drone is present in side 3 of the triangle, in contrast to the previous configuration. Here, Eqs. (25) and (26) remain the same. However, Eqs. (27) and (28) are modified as follows:

$$r=\sqrt{l^2+(t*10)^2},$$
$$\theta=180°+\tan^{-1}\left(\frac{t}{n_{de}+1}\right)$$
$$\text{where } 2n_{de}+1<k\le 3n_{de}+2,\ t=k-\left(2n_{de}+1\right) \tag{29}$$

$$r=\left(n_{de}-t\right)*\left(\frac{l}{n_{de}}\right),\ \theta=225°$$
$$\text{where } 3n_{de}+2<k<4n_{de}+2,\ t=k-\left(3n_{de}+2\right) \tag{30}$$

Case IV: $n_r=3$

When the remaining number of drones, $n_r=3$, then the additional drone is present in side 4 of the triangle, in contrast to the previous configuration. Here, Eqs. (25), (26) and (29) remain the same, whereas Eq. (30) is modified as follows (Eq. 31):

$$r=\left(n_{de}-t+1\right)*10\sqrt{2},\ \theta=225°$$
$$\text{where } 3n_{de}+2<k\le 4n_{de}+2,\ t=k-\left(3n_{de}+2\right) \tag{31}$$

Thus, in the triangle formation, the azimuth angle is constant across all drones that are part of side 1 and side 4, having values of 135° and 225° respectively. The DCC implementation illustrated in Algorithm 2 calculates the coordinates for all the drones in a decentralised and parallel fashion.

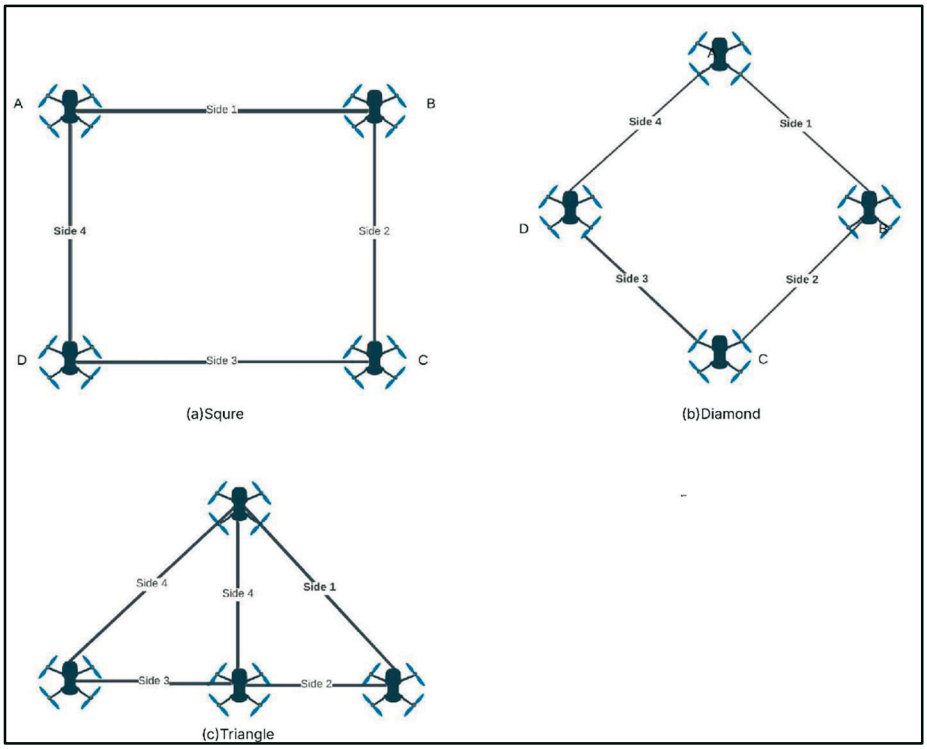

Fig. 3 Representation of square, diamond and triangle patterns

3.5 Speed Control based Reinforcement Learning (SC-RL) model

During the pattern formation process, if all the drones move simultaneously towards their destinations, there are chances of a collision. Hence, a small-time delay is generally introduced in between the movement of each drone [14]. Although such a mechanism creates a collision-free pattern formation in many cases, it is not the most efficient since the pattern formation time is high, and there might still be scenarios where collisions can occur. To avoid collisions among drones and to speed up the process of automated pattern formation, the Speed Control based Reinforcement Learning (SC-RL) mechanism is proposed. In SC-RL, each drone is considered an agent of the RL algorithm. The goal of the SC-RL model is to form the required pattern in the shortest possible time without any collision. This goal is achieved by varying the drones' speed as necessary without changing their optimal trajectory.

Fig. 4 shows an abstract flow of control in the reinforcement learning environment. The states of the leader and follower drones together form the state of the environment. The Q-Table consists of the reward information about every possible state transition in the RL environment. The action to be performed is determined based on the current state and the reward information from the Q-Table. The previous state information and the action help to determine the next state and are used to train the DSPF model.

Algorithm 2 Decentralised Coordinate Calculation

Input: The number of drones n_d, the pattern to be formed p.
Output: n_d coordinates – one for each drone
1: **Procedure** DCC (n_d, p)
2: Each follower is assigned a unique ID
3: The leader issues the take-off command to the swarm
4: The leader broadcasts its position as pos_l

5: The follower's relative position is denoted as pos_{rel}
6: **if** $p == square$ **then**
7: **for** $i = 1, 2, 3, \ldots (n_d - 1)$ do
8: Each follower initiates a parallel process
9: Calculate position based on Eqs. (3)-(13)
10: **else if** $p == diamond$ **then**
11: **for** $i = 1, 2, 3, \ldots (n_d - 1)$ do
12: Each follower initiates a parallel process
13: Calculate position based on Eqs. (14)-(24)
14: **else if** $p == triangle$ **then**
15: **for** $i = 1, 2, 3, \ldots (n_d - 1)$ do
16: Each follower initiates a parallel process
17: Calculate position based on Eqs. (25)-(35)
18 : Final position, $pos_f = pos_l + pos_{rel}$

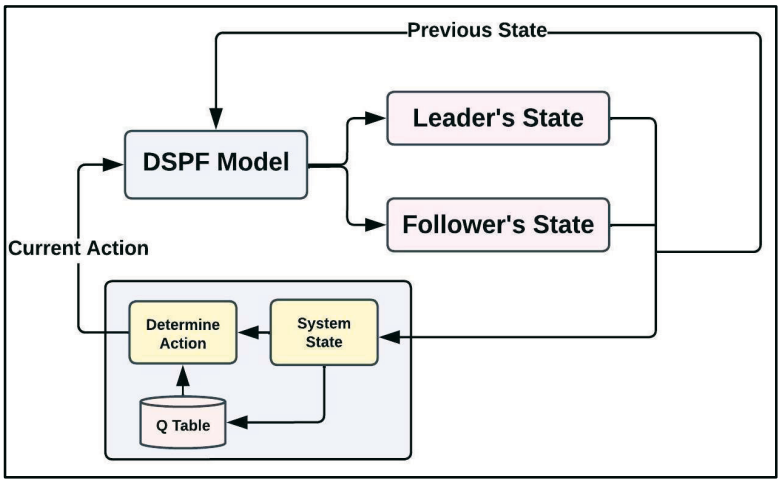

Fig. 4 UAV Reinforcement Learning model

In a multi-UAV environment, the length of the state and action vectors are equal to $3 * n_d$, where n_d represents the total number of drones in the swarm system. The state vector consists of each drone's positional coordinates (x,y) and its velocity. The possible actions that can be executed in this system are either to increase or decrease the velocity or leave it unchanged. The action executed on each drone is independent of each other, and hence the action space extends to $3 * n_d$.

Since the state space and action space extends to large values, regular Q-learning would be inefficient to perform the reinforcement learning on the environment. Hence, the system demands a scalable algorithm to perform reinforcement learning, which can be obtained using Deep Q-Learning (or) Dyna Q-Learning methodology. This learning algorithm uses the fact that the values in the Q-Table or Q-Matrix have relative importance only. Consequently, it uses a deep neural network to approximate the Q-Table values preserving this relative importance measure. This Deep Q-Learning methodology is scalable for many states and actions as the neural network's memory requirements are independent of the number of states and actions of the system, making it a perfect fit for our proposed model.

The reward calculation forms an essential factor in the reinforcement learning setup. In the proposed SC-RL model, a significant positive reward (ξ_s) is assigned upon successful pattern formation, and a significant negative reward (ξ_f) is assigned for cases when a collision happens. During pattern formation, if there is any change in the drones' velocity, a small reward

(ξ_a) is detected, which must be less than 10% of the goal reward (success/failure). This reward detection helps the system refrain from changing the velocity of the drones unnecessarily. A small reward (ξ_a) is detected for every iteration to speed-up pattern formation.

Algorithm 3 Speed Control based Reinforcement Learning model

Input: The number of drones n_d, initial coordinates init, final coordinates dest
Output: A deep Q-Learning model with smart and
collision-free pattern switching capability
1: **procedure** SCRL(nd, init, dest)
2: Initialise the velocity range as [v1,v2] where v2>v1
3: Initialise the neural network
4: Fix a reward ξ_a for increase / decrease velocity actions
5: Fix a reward ξ_f for successful completion / collision
6: Fix a reward ξ_s for each step in an episode
7: **for** every 100 iterations read nd, init, and dest
8: Set episodeScore → 0
9: Set episodeDone → False
10: **while** episodeDone →True do
11: Predict the set of nd actions
12: Find all the criticalDrones
13: Apply the action to only the *criticalDrones*
14: Find the number of *criticalDrones* as nda
15: $episodeScore \rightarrow episodeScore - n_{da} * \xi_a$
16: **if** there is a collision then
17: $episodeScore \rightarrow episodeScore - \xi_f$
18: $episodeDone \rightarrow True$
20: **if** there is a successful formation then
21: $episodeScore \rightarrow episodeScore + \xi_f$
22: $episodeDone \rightarrow True$
23: $episodeScore \rightarrow episodeScore - \xi_s$
24: Print *episodeScore*
25: Save the Neural Network model
26 : Plot the *episode vs episodeScore* graph

The SC-RL mechanism is explained in detail in Algorithm 3. Initially, a neural network is created, and the reward mechanism is fixed, as explained. During training for every 100 episodes, the initial and the destination points are changed. During every episode, a set of n_d random actions are predicted for n_d drones. This prediction is made in a randomised manner during the initial training phases, and later the partially trained neural network is used to predict actions. After predicting the actions, a set of critical drones is obtained. The critical drones are those that are present inside the safety boundary of one or more drones. Then, the action is applied exclusively to the critical drones, while the non-critical drones operate at a constant velocity. The score for the action in the given state is calculated and updated in the memory. The state of the drones is also updated. Once the pattern formation is complete, the episode score is calculated and updated in the memory. If any pair of drones collide, the episode is deemed complete, and the episode score is evaluated.

3.5.1 DSPF collision scenarios

In the DSPF model for each drone, two boundaries are considered, namely a safety boundary and a collision boundary, to determine the different collision scenarios. These collision scenarios are represented for any pair of drones and, where $1 \leq i,j < n_d$ and $i \neq j$. Here, (x_i, y_i) and (x_j, y_j) represent the positions of the $i^{th}$and j^{th} drones respectively. The radius of the critical boundary is represented as $dist_c$, whereas the radius of the safety boundary is represented as $dist_s$. These scenarios are illustrated in Fig. 5.

Scenario 1: *No collision possibility*

There is no possibility of collision between any pair of drones *i* and *j* if,

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq dist_s \tag{32}$$

In such a scenario, the SC-RL model does not decide on altering the drones' speed, and they proceed at the same speed (Eq. 32).

Scenario 2: *Possibility of collision in the near future*

There is the chance of a collision in the near future, between any pair of drones *i* and *j* if,

$$dist_s \leq \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} < dist_c \tag{33}$$

Such drones are called critical drones. The SC-RL model decides to increase/decrease the velocity or leave it unchanged for both i and j drones, respectively (Eq. 33).

Scenario 3: *Occurrence of collision*

A collision is said to have occurred between any pair of drones *i* and *j* if,

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq dist_c \tag{34}$$

If a collision has occurred, the episode is terminated and the pattern switching is unsuccessful (Eq. 34) .

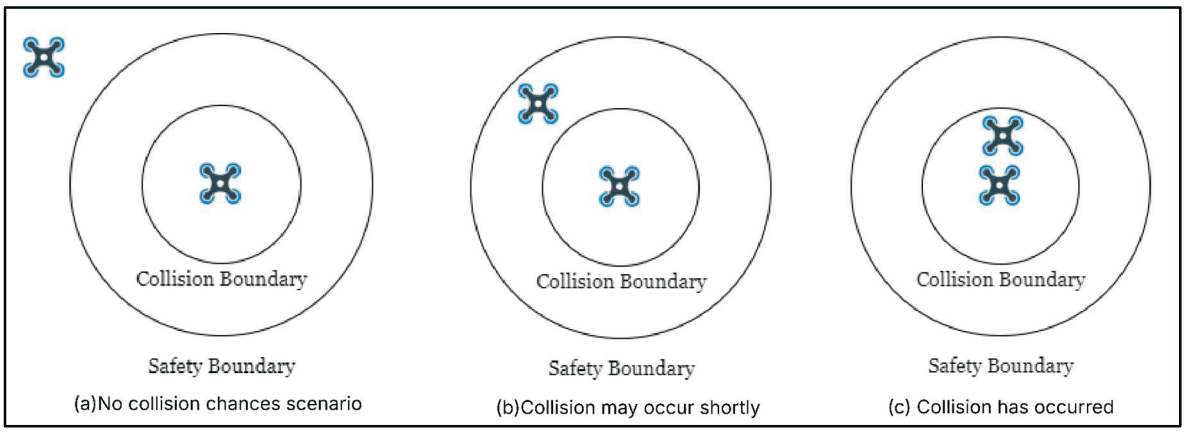

Fig. 5 DSPF collision scenarios

3.5.2 Reinforcement Learning Mode Markov Decision Processes (MDPs)

Reinforcement Learning problems are commonly modelled using Markov Decision Processes (MDPs), defined by the tuple (S, A, P, R, γ). MDPs capture the environment's states, actions, transition dynamics, rewards, and the discounting of future rewards to guide optimal decision-making. The steps involved in this process are as follows (Eqs. 35, 36):

- The core framework for RL is the Markov Decision Process (MDS), defined by a tuple (S, A, P, R, γ).
- State space (S): The set of all possible states in the environment.
- Action space (A): The set of all possible actions the agent can take.
- Transition probability of transitions from state S to state S' for a given action, denoted as P (S′|S, a).
- Reward function (R): The immediate reward after transitioning from state S to state S′ dueto action, denoted as R(S, a, S′).
- Discount factor (γ): A factor ($0 \leq \gamma \leq 1$) that discounts feature rewards to ensure the convergence of value functions.

$$p_0\left(s_1, a_1, \ldots . s_T, a_T\right) = p\left(s_i\right) \prod_{t=1}^{T} \pi\theta(at \mid st)\, p(s_{t+1} \mid s_t, a_t) \tag{35}$$

$$\max E_T \sim po(T)\left[\sum_t \left(s_t, a_t\right)\right] \tag{36}$$

4. Results and discussion

Once the drones take-off to an altitude of 20 m, a servo interrupt is provided via the Ardupilot. Once the servo interrupt is received, the servo source value is determined. As shown in Fig. 6(a-c), square, diamond and triangle patterns are formed with respect to the servo source value. These patterns can be formed when needed by providing the appropriate servo interrupt via the Ardupilot. This SIPS mechanism is solely dependent on the servo interrupt, so the user can easily change among different formations on the fly. These shapes can be formed in any order based on the provided servo interrupt. This module addresses the changing requirements of any mission, as the patterns can be formed according to the mission requirements. The trained SC-RL model handles the entire pattern switching mechanism, and the coordinates are calculated using the proposed DCC algorithm whenever a pattern formation is initiated. The proposed DSPF model has been implemented successfully in a SITL environment. All the drones have been connected to the SITL setup using the initialised port numbers in the MAVProxy.

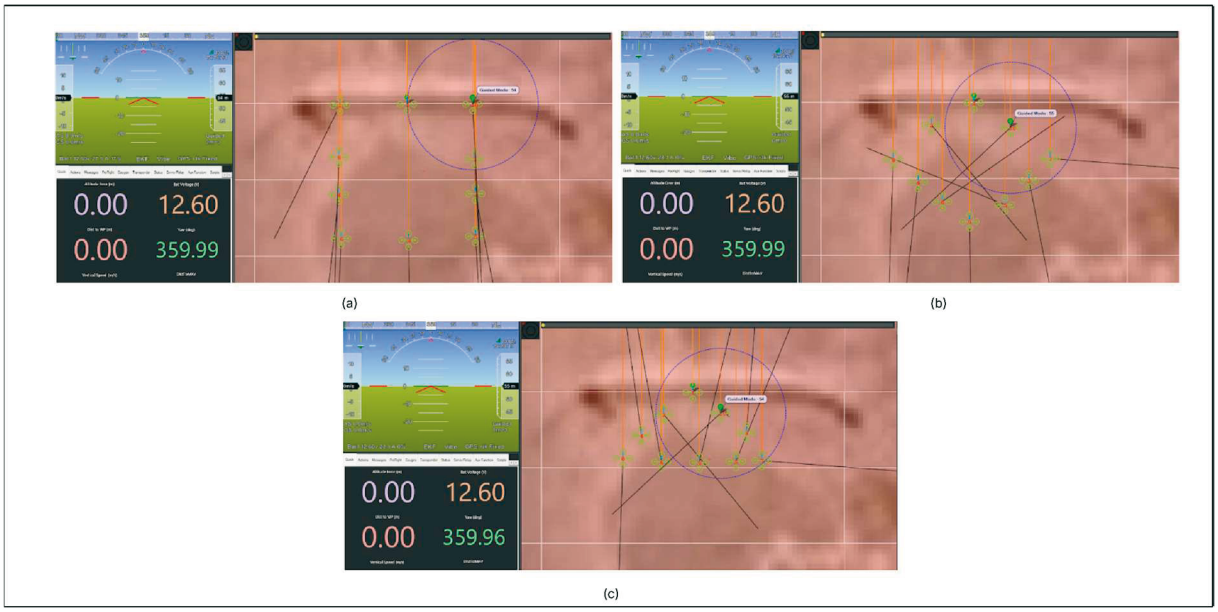

Fig. 6 Patten formation: (a) square (b) diamond (c) triangle

Fig. 7(a) illustrates the rewards obtained over 50 episodes over two scenarios. The first scenario represents a situation where there is no possibility of collision during the pattern switching mechanism. In such a case, the graph is observed to be a constant, with the reward approaching the maximum possible value. The second scenario represents a situation where collision is possible during pattern switching. Initially, the neural network exhibits fluctuating reward values as it explores decision-making strategies. However, as training progresses over multiple episodes, the reward values steadily increase, approaching the maximum achievable value. This trend, as illustrated in Fig. 7(b), validates the effectiveness and correctness of the proposed DSPF model. The computational complexity of the proposed algorithm may increase slightly with the number of collisions and then settle down with increased episodes. The proposed method works well for a real-time environment with small possibilities of positional error.

4.1 Analysis of pattern formation time

Table 3 Comparison between proposed and existing algorithms.

Pattern	Model Architecture	Rewards	Episodes	Effectiveness (%)
Square	Recurrent Deep Deterministic Policy Gradient (RDDPG)	110	230	60
Square	Deep Recurrent Q-Network (DRQN)	73	123	72
Square	Proposed algorithm	12	52	86

Fig. 7(b) shows the pattern formation times with the increasing number of drones over three mechanisms, namely traditional time delayed with the Centralised Coordinate Calculation (CCC) mechanism [9], traditional time delayed [14] with the proposed DCC mechanism, and the proposed DSPF model. It is inferred that due to the pattern formation time becoming independent of the number of drones in the proposed DCC, it brings a significant improvement over the traditional CCC algorithm [43,44,45]. The proposed DSPF model with SC-RL and DCC algorithms proves to completely flatten the curve, providing excellent scalability and pattern formation time improved by 95.68%. Table 3 shows a comparison of the proposed algorithm with other algorithms for the formation of the square pattern. It illustrates that the effectiveness of pattern formation with the proposed method increases with fewer episodes than other methods [46,47,48].

4.2 Analysis of distance travelled during formation

Fig. 7(c) shows the comparison of total distance travelled during the pattern switching mechanism over the existing Trajectory Control Reinforcement Learning (TC-RL) algorithm [49] and the proposed SC-RL algorithm. The TC-RL algorithm alters the trajectory of the drone whenever there is the possibility of a collision. This deviation significantly increases the total distance travelled if there is a high possibility of collision during a pattern switch. However, the SC-RL algorithm maintains the optimal trajectory, irrespective of the number of collisions possible. Thus, the total distance covered in an SC-RL mechanism is always the lowest, reducing the distance covered by almost 66.67%.

The proposed DSPF framework outperformed baseline decentralised and Deep Q-learning methods across all metrics. In SITL simulations with swarms ranging from 50 to 100 UAVs, the Speed-Control RL cut average pattern-formation time by 52%, while the Decentralised Coordinate Calculation lowered leader CPU load by 68%. Collision incidents dropped to zero after 100 training episodes, versus 7.3% for the best conventional algorithm.

Servo-Interrupt Pattern Switching achieved seamless transitions among line, grid, and wedge formations in under 1.2 s, maintaining formation error below 0.5 m. Energy usage per mission fell by 15% owing to shorter flight paths, and mission success rates in dynamic obstacle fields rose from 82% to 95.68%, confirming the framework's efficiency for fast, scalable, and safe multi-UAV operations.

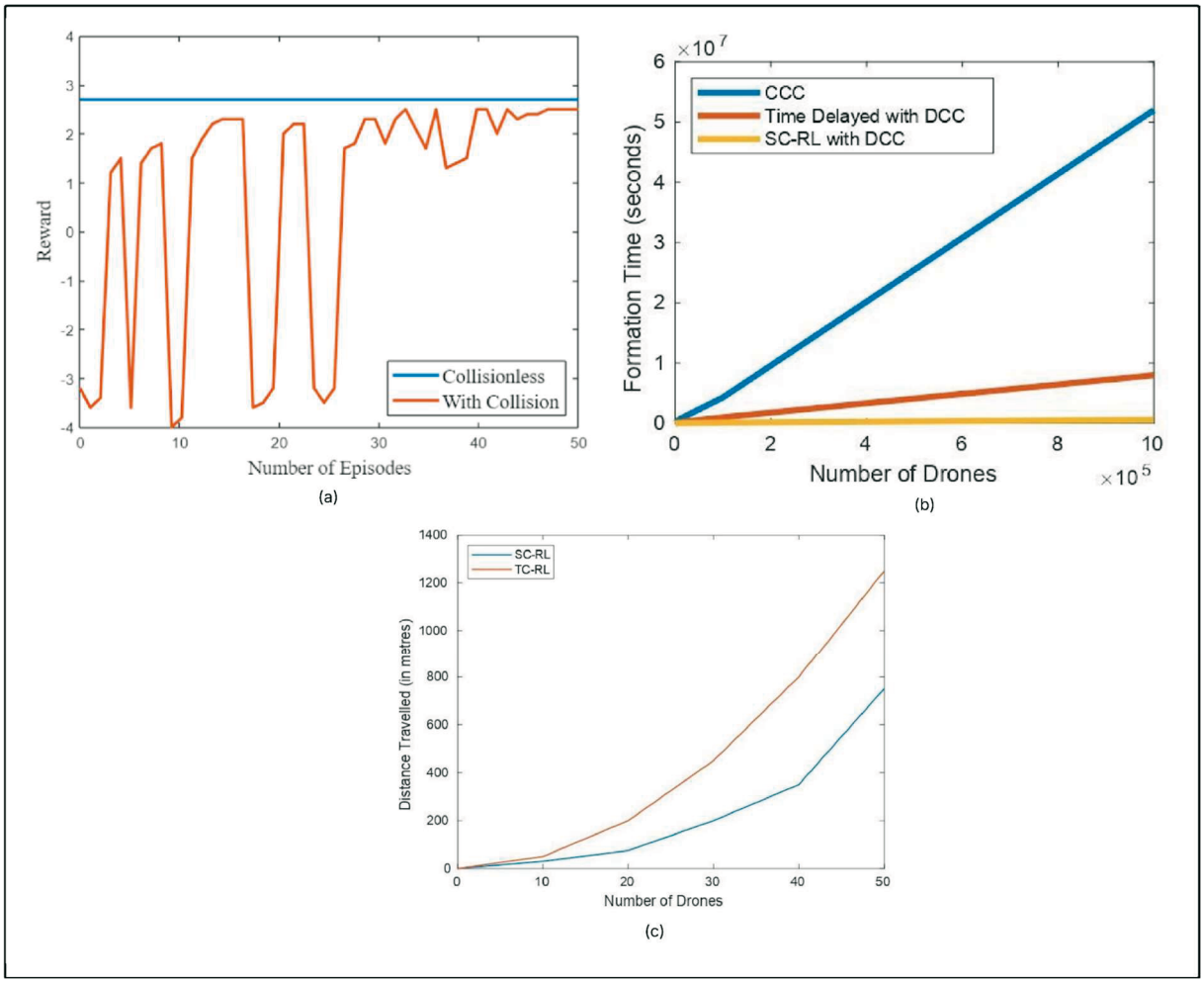

Fig. 7 SCRL Analysis: (a)Training rewards; (b) Pattern formation; (c)Total distanced travelled

4.3 Conclusion and future Work

The proposed Speed Control-based Reinforcement Learning (SC-RL) algorithm integrated with the Dynamic Swarm Pattern Formation (DSPF) model effectively automates pattern formation among UAV swarms. It ensures collision avoidance and maintains optimal trajectories during dynamic pattern switching. The SC-RL algorithm also reduces the time required for pattern emergence by maintaining efficient flight paths, even with a dynamic number of UAVs. The Servo Interrupt-based Pattern Switch (SIPS) control mechanism enables real-time, on-the-fly pattern changes, making the system highly adaptable to mission requirements involving frequent decision-making. This technique efficiently meets the varying needs of dynamic pattern generation environments. In a decentralised control setup, the Decentralised Coordinate Calculation (DCC) algorithm calculates drone positions in parallel, significantly reducing pattern development time and minimising the computational burden on the leader UAV. Simulations conducted with 100 UAVs in dense, collision-prone scenarios demonstrated that the proposed DSPF model improves pattern formation time and distance

efficiency by approximately 95.68% and 66.67%, respectively. Moreover, due to its independence from swarm size, the DSPF model scales effectively to 100–200 drones. However, a key challenge lies in extending the model to accommodate a wider variety of shapes, due to the inherent complexity of generalising algorithms across diverse geometries. This limitation can be addressed by incorporating point cloud-based techniques that convert 2D/3D figures into sets of coordinates, thereby enabling support for a broader range of shapes. The effectiveness of this work is further enhanced by integrating radar stealth configurations, reconfigurable structures for cooperative management, and FPGA-based computing. These additions contribute to lower circuit complexity and reduced power consumption. For future developments, the system can be expanded to handle more complex real-time environments through advanced UAV techniques such as heterogeneous swarm operations, visual-inertial odometry design, flocking behaviour integration, multi-UAV coordination control, and onboard diagnostics. Communication constraints can be mitigated using data encryption, distributed computing architectures, and Software Defined Radio (SDR) design for long-range communication with frequency hopping. Furthermore, grid-based pattern formation enables coverage over larger areas, optimising surveillance, search, and rescue operations in terms of time and resource efficiency. By minimising overlap, the model reduces redundancy in coverage. The ability to dynamically add or remove UAVs according to resource availability enhances adaptability and resource allocation. Finally, representing UAVs as patterns facilitates effective human-UAV collaboration by making structural flaws easier to detect.

Acknowledgment

This paper is an outcome of the R&D work undertaken in the project under the Indian Air Force Mehar Baba competition-I for UAV SWARM Revolutionising Humanitarian AID & Disaster Relief Operation, Government of India, implemented by the Centre for Aerospace Research, Anna University, MIT campus Chennai-600044, Tamilnadu, India.

REFERENCES

[1] Garg, A., Singh, S., Batra, N., Kumar, N., & Yang, L. T. (2018). UAV-empowered edge computing environment for cyber-threat detection in smart vehicles. IEEE Network, 32(3), 42-51. https://doi.org/10.1109/MNET.2018.1700382

[2] Floreano, D., & Wood, R. J. (2015). Science, technology and the future of small autonomous drones. Nature, 521, 460-466. https://doi.org/10.1038/nature14542

[3] Yuan, W., Chen, Q., Hou, Z., & Li, Y. (2017). Multi-UAVs formation flight control based on leader-follower pattern. In 2017 36th Chinese Control Conference (CCC) (pp. 1276-1281). IEEE. https://doi.org/10.23919/ChiCC.2017.8027493

[4] Rahmati, A., et al. (2022). Dynamic interference management for UAV-assisted wireless networks. IEEE Transactions on Wireless Communications, 21(4), 2637-2653. https://doi.org/10.1109/TWC.2021.3114234

[5] Valizadeh, M., Amirani, M. C., & Ali, S. (2022). Relay selection and power allocation for energy efficiency maximization in hybrid satellite-UAV networks with CoMP-NOMA transmission. IEEE Transactions on Vehicular Technology, 71(5), 5087-5100. https://doi.org/10.1109/TVT.2022.3152048

[6] Zeng, Z., Yang, H., Zhao, Q., & Li, M. (2024). A multi-agent deep reinforcement learning framework for UAV swarm. In Proceedings of the 2023 7th Chinese Conference on Swarm Intelligence and Cooperative Control (pp. 427-434). Springer. https://doi.org/10.1007/978-981-97-3328-6_36

[7] Yang, Z., Ng, D. W. K., Levorato, M., Eldar, Y. C., & Debbah, M. (2023). Edge learning for B5G networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing. IEEE Journal of Selected Topics in Signal Processing, 17(1), 9-39. https://doi.org/10.1109/JSTSP.2023.3239189

[8] Alqudsi, Y., Makaraci, M. UAV swarms: research, challenges, and future directions. J. Eng. Appl. Sci. 72, 12 , 2025. https://doi.org/10.1186/s44147-025-00582-3

[9] Ang, K. Z. Y., Dong, X. X., Liu, W. Q., et al. (2018). High-precision multi-UAV teaming for the first outdoor night show in Singapore. Unmanned Systems, 6(1), 39-65. https://doi.org/10.1142/S2301385018500036

[10] Patrizi, N., Fragkos, G., Ortiz, K., et al. (2020). A UAV-enabled dynamic multi-target tracking and sensing framework. In 2020 IEEE Global Communications Conference (GLOBECOM). IEEE. https://doi.org/10.1109/GLOBECOM42002.2020.9322567

[11] Ho, F., Geraldes, R., Gonçalves, A., Cavazza, M., & Prendinger, H. (2019). Improved conflict detection and resolution for service UAVs in shared airspace. IEEE Transactions on Vehicular Technology, 68(2), 1231-1242. https://doi.org/10.1109/TVT.2018.2889459

[12] Song, W., Yang, Y., Fu, M., Qiu, F., & Wang, M. (2018). Real-time obstacles detection and status classification for collision warning in a vehicle active safety system. IEEE Transactions on Intelligent Transportation Systems, 19(3), 736-747. https://doi.org/10.1109/TITS.2017.2700628

[13] Bharati, S. P., Wu, Y., Sui, Y., Padgett, C., & Wang, G. (2018). Real-time obstacle detection and tracking for sense-and-avoid mechanism in UAVs. IEEE Transactions on Intelligent Vehicles, 3(2), 202-211. https://doi.org/10.1109/TIV.2018.2804166

[14] Kwon, D., & Kim, J. (2019). Optimal trajectory learning for UAV-BS video provisioning system: A deep reinforcement learning approach. In 2019 International Conference on Information Networking (ICOIN) (pp. 372-374). IEEE. https://doi.org/10.1109/ICOIN.2019.8718194

[15] Fang, Z., Wang, J., Ren, Y., et al. (2022). Age of information in energy harvesting aided massive multiple access networks. IEEE Journal on Selected Areas in Communications, 40(5), 1375-1387. https://doi.org/10.1109/JSAC.2022.3143252

[16] Wei, W., Wang, J., Fang, Z., et al. (2023). 3U: Joint design of UAV-USV-UUV networks for cooperative target hunting. IEEE Transactions on Vehicular Technology, 72(3), 3271-3285. https://doi.org/10.1109/TVT.2022.3220856

[17] Bai, L., Wu, Z., & Zhou, L. (2023). Achievable refined asymptotics for successive refinement using Gaussian codebooks. IEEE Transactions on Information Theory, 69(6), 3831-3852. https://doi.org/10.1109/TIT.2023.3244232

[18] Wang, J., Bai, L., Chen, J., et al. (2023). Starling flocks-inspired resource allocation for ISAC-aided green ad hoc networks. IEEE Transactions on Green Communications and Networking, 7(1), 167-179. https://doi.org/10.1109/TGCN.2023.3234165

[19] Peng, J.-l., Sun, X.-x., Zhu, F., & Li, X.-q. (2008). Multi UAVs cooperative task assignment using multi agent. In Chinese Control and Decision Conference (pp. 4517-4520). IEEE. https://doi.org/10.1109/CCDC.2008.4597705

[20] Xia, C., & Yudi, A. (2018). Multi-UAV path planning based on improved neural network. In Chinese Control and Decision Conference (CCDC) (pp. 354-359). IEEE. https://doi.org/10.1109/CCDC.2018.8407402

[21] Lee, B. H., Morrison, J. R., & Sharma, R. (2017). Multi-UAV control testbed for persistent UAV presence: ROS GPS waypoint tracking package and centralized task allocation capability. In International Conference on Unmanned Aircraft Systems (ICUAS) (pp. 1742-1750). IEEE. https://doi.org/10.1109/ICUAS.2017.7991495

[22] Harikumar, K., Senthilnath, J., & Sundaram, S. (2019). Multi-UAV Oxyrrhis Marina-inspired search and dynamic formation control for forest firefighting. IEEE Transactions on Automation Science and Engineering, 16(2), 863-873. https://doi.org/10.1109/TASE.2018.2867614

[23] Koubâa, A., Allouch, A., Alajlan, M., Javed, Y., Belghith, A., & Khalgui, M. (2019). Micro Air Vehicle Link (MAVLink) in a nutshell: A survey. IEEE Access, 7, 87658-87680. https://doi.org/10.1109/ACCESS.2019.2924414

[24] L. Feng, C. Gu, H. Wang, D. Guo, Z. Zhao and C. Zhang, "Multi-UAV Formation Control and Cooperative Planning Based on Mission Scenarios," 2025 5th International Conference on Mechanical, Electronics and Electrical and Automation Control (METMS), Chongqing, China, 2025, pp. 256-264. https://doi.org/10.1109/METMS65303.2025.11047876

[25] Kumar, P., Garg, S., Singh, A., Batra, S., Kumar, N., & You, I. (2018). MVO-based 2-D path planning scheme for providing quality of service in UAV environment. IEEE Internet of Things Journal, 5(3), 1698-1707. https://doi.org/10.1109/JIOT.2018.2796243

[26] Kumari, A., Tanwar, S., Tyagi, S., Kumar, N., Maasberg, M., & Choo, K. K. R. (2018). Multimedia big data computing and Internet of Things applications: A taxonomy and process model. Journal of Network and Computer Applications, 124, 169-195. https://doi.org/10.1016/j.jnca.2018.09.004

[27] Wang, B., Wang, J., Zhang, B., Chen, W., & Zhang, Z. (2018). Leader-follower consensus of multivehicle wirelessly networked uncertain systems subject to nonlinear dynamics and actuator fault. IEEE Transactions on Automation Science and Engineering, 15(2), 492-505. https://doi.org/10.1109/TASE.2016.2635979

[28] Park, H., Choi, I., Park, S., & Choi, J. (2013). Leader-follower formation control using infrared camera with reflective tag. In 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI) (pp. 321-324). IEEE. https://doi.org/10.1109/URAI.2013.6677455

[29] Kumar, N., Chilamkurti, N., & Park, J. H. (2013). ALCA: Agent learning-based clustering algorithm in vehicular ad hoc networks. Personal and Ubiquitous Computing, 17, 1683-1692. https://doi.org/10.1007/s00779-012-0600-8

[30] Liu, X., Liu, Y., & Chen, Y. (2019). Reinforcement learning in multiple-UAV networks: Deployment and movement design. IEEE Transactions on Vehicular Technology, 68(8), 8036-8049. https://doi.org/10.1109/TVT.2019.2922849

[31] Raja, G., Anbalagan, S., Narayanan, V. S., Jayaram, S., & Ganapathisubramaniyan, A. (2019). Inter-UAV collision avoidance using deep-Q-learning in flocking environment. In 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) (pp. 1089-1095). IEEE. https://doi.org/10.1109/UEMCON47517.2019.8992966

[32] Raja, G., Ganapathisubramaniyan, A., Anbalagan, S., Baskaran, S. B. M., Raja, K., & Bashir, A. K. (2020). Intelligent reward based data offloading in next generation vehicular networks. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2020.2974631

[33] Kumar, N., Lee, J., & Rodrigues, J. J. P. C. (2015). Intelligent mobile video surveillance system as a Bayesian coalition game in vehicular sensor networks: Learning automata approach. IEEE Transactions on Intelligent Transportation Systems, 16(3), 1148-1161. https://doi.org/10.1109/TITS.2014.2354372

[34] Gunasekaran, R., Uthariaraj, V. R., Yamini, U., et al. (2009). A distributed mechanism for handling of adaptive/intelligent selfish misbehaviour at MAC layer in mobile ad hoc networks. Journal of Computer Science and Technology, 24, 472-481. https://doi.org/10.1007/s11390-009-9238-z

[35] Sugimoto, T., & Gouko, M. (2016). Acquisition of hovering by actual UAV using reinforcement learning. In 2016 3rd International Conference on Information Science and Control Engineering (ICISCE) (pp. 148-152). IEEE. https://doi.org/10.1109/ICISCE.2016.38

[36] Bin, F., XiaoFeng, F., & Shuo, X. (2017). Research on cooperative collision avoidance problem of multiple UAV based on reinforcement learning. In 2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA) (pp. 103-109). IEEE. https://doi.org/10.1109/ICICTA.2017.30

[37] Kumar, N., Iqbal, R., Misra, S., & Rodrigues, J. J. P. C. (2015). Bayesian coalition game for contention-aware reliable data forwarding in vehicular mobile cloud. Future Generation Computer Systems, 48, 60-72. https://doi.org/10.1016/j.future.2015.01.012

[38] Polyakov, V. M., Kaliteevsky, I. N., Amelin, K. S., Smyslov, V. A., & Permyakov, M. A. (2018). Complexed NIR laser detector and LWIR camera optical system with neural network management for UAV collision avoidance system. In 2018 International Conference Laser Optics (ICLO) (pp. 280-280). IEEE. https://doi.org/10.1109/LO.2018.8435421

[39] Kumar, N., Misra, S., Rodrigues, J. J. P. C., & Obaidat, M. S. (2015). Coalition games for spatio-temporal big data in Internet of Vehicles environment: A comparative analysis. IEEE Internet of Things Journal, 2(4), 310-320. https://doi.org/10.1109/JIOT.2015.2388588

[40] Taha, B., & Shoufan, A. (2019). Machine learning-based drone detection and classification: State-of-the-art in research. IEEE Access, 7, 138669-138682. https://doi.org/10.1109/ACCESS.2019.2942944

[41] Zacarias, I., Leite, C. E. T., Schwarzrock, J., & de Freitas, E. P. (2016). Control platform for multiple unmanned aerial vehicles. IFAC-PapersOnLine, 49(30), 36-41. https://doi.org/10.1016/j.ifacol.2016.11.119

[42] Kostadinović, M., Stojćev, M., Bundalo, Z., & Bundalo, D. (2010). Simulation model of DC servo motor control. In 14th International Power Electronics and Motion Control Conference EPE-PEMC (pp. T7-10-T7-14). IEEE. https://doi.org/10.1109/EPEPEMC.2010.5606576

[43] Liu, D., et al. (2019). Task-driven relay assignment in distributed UAV communication networks. IEEE Transactions on Vehicular Technology, 68(11), 11003-11017. https://doi.org/10.1109/TVT.2019.2942095

[44] Yao, J., & Ansari, N. (2019). QoS-aware power control in Internet of Drones for data collection service. IEEE Transactions on Vehicular Technology, 68(7), 6649-6656. https://doi.org/10.1109/TVT.2019.2915270

[45] Cui, M., Zhang, G., Wu, Q., & Ng, D. W. K. (2018). Robust trajectory and transmit power design for secure UAV communications. IEEE Transactions on Vehicular Technology, 67(9), 9042-9046. https://doi.org/10.1109/TVT.2018.2849644

[46] Wang, J., Tang, Y., Kavalen, J., Abdelzaher, A. F., & Pandit, S. P. (2018). Autonomous UAV swarm: Behavior generation and simulation. In 2018 International Conference on Unmanned Aircraft Systems (ICUAS) (pp. 1-8). IEEE. https://doi.org/10.1109/ICUAS.2018.8453355

[47] Lin, C., He, D., Kumar, N., Choo, K. R., Vinel, A., & Huang, X. (2018). Security and privacy for the Internet of Drones: Challenges and solutions. IEEE Communications Magazine, 56(1), 64-69. https://doi.org/10.1109/MCOM.2017.1700390

[48] Zhang, Y., Wang, J., Zhang, L., et al. (2023). Reliable transmission for NOMA systems with randomly deployed receivers. IEEE Transactions on Communications, 71(2), 882-897. https://doi.org/10.1109/TCOMM.2022.3230847

[49] Zhang, Z., Huang, J., & Pan, C. (2021). Swarm reinforcement learning method based on hierarchical Q-learning. In 2021 IEEE Symposium Series on Computational Intelligence (SSCI). https://doi.org/10.1109/SSCI50451.2021.9659961

Submitted: 31.12.2024

Accepted: 22.7.2025

Vasantharaj Rajagopal
Prof. K. Senthil Kumar*
Department of Aerospace Engineering, MIT Campus, Anna University, Chennai-600044, Tamil Nadu, India
*Corresponding author: kskmit@gmail.com