Expert Systems With Applications 235 (2024) 121240

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Q-Learning based system for Path Planning with Unmanned Aerial Vehicles swarms in obstacle environments

Alejandro Puente-Castro [a,*], Daniel Rivero [a], Eurico Pedrosa [b], Artur Pereira [b], Nuno Lau [b], Enrique Fernandez-Blanco [a]

[a] *Faculty of Computer Science, CITIC, University of A Coruna, A Coruna, 15007, Spain*
[b] *IEETA, DESI, LASI, University of Aveiro, Portugal*

ARTICLE INFO

Keywords:
UAV
Swarm
Obstacle
Path Planning
Reinforcement learning
Artificial Neural Network

ABSTRACT

Path Planning methods for the autonomous control of Unmanned Aerial Vehicle (UAV) swarms are on the rise due to the numerous advantages they bring. There are increasingly more scenarios where autonomous control of multiple UAVs is required. Most of these scenarios involve a large number of obstacles, such as power lines or trees. Despite these challenges, there are also several advantages; if all UAVs can operate autonomously, personnel expenses can be reduced. Additionally, if their flight paths are optimized, energy consumption is reduced, leaving more battery time for other operations. In this paper, a Reinforcement Learning-based system is proposed to solve this problem in environments with obstacles by utilizing Q-Learning. This method allows a model, in this case, an Artificial Neural Network, to self-adjust by learning from its mistakes and successes. Regardless of the map's size or the number of UAVs in the swarm, the goal of these paths is to ensure complete coverage of an area with fixed obstacles for tasks like field prospecting. Setting goals or having any prior information apart from the provided map is not required. During the experimentation phase, five maps of varying sizes were used, each with different obstacles and a varying number of UAVs. To evaluate the quality of the results, the number of actions taken by each UAV to complete the task in each experiment was considered. The results indicate that the system achieves solutions with fewer movements as the number of UAVs increases. An increasing number of UAVs on a map lead to solutions in fewer moves. The results have been compared, and a statistical significance analysis has been conducted on the proposed model's outcomes, demonstrating its capabilities. Thus, it is shown that a two-layer Artificial Neural Network used to implement a Q-Learning algorithm is sufficient to operate on maps with obstacles.

1. Introduction

New uses for swarms of unmanned aerial vehicles (UAVs) are being developed to solve different industrial and emergency problems (Albani, IJsselmuiden, Haken, & Trianni, 2017; Bocchino, Canham, Watney, Reder, & Levison, 2018; Corte et al., 2020; Huuskonen & Oksanen, 2018; Liu, Wang, Wang, Shu, & Li, 2018; Rabinovitch, Lorenz, Slimko, & Wang, 2021). The advantages provided by UAVs, such as their low cost, excellent mobility, safety, and convenient size for some maneuvers, are the main reasons for their growing popularity (Yeaman & Yeaman, 1998). All these advantages are offered by the wide variety of UAVs that exist to fulfill every need. This variety allows for the integration of different types of sensors with varying capabilities. The development of sensors for UAVs is on the rise, particularly in remote sensing (Noor, Abdullah, & Hashim, 2018). The flexibility in the characteristics of UAVs, such as their architecture or the sensors they accommodate, makes them popular tools for diverse needs (Austin, 2011).

However, UAVs have drawbacks, with the most significant one being power consumption, which reduces operational time. Due to their small size, it is challenging to acquire compact power sources with substantial capacities while also keeping the weight low. When the weight is minimal, flight operations benefit from extended flight time availability.

The limitations on flight time imposed by batteries can be mitigated when groups or swarms are employed. In essence, as flight paths

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

[*] Corresponding author.

E-mail addresses: a.puentec@udc.es (A. Puente-Castro), daniel.rivero@udc.es (D. Rivero), efp@ua.pt (E. Pedrosa), artur@ua.pt (A. Pereira), nunolau@ua.pt (N. Lau), enrique.fernandez@udc.es (E. Fernandez-Blanco).

https://doi.org/10.1016/j.eswa.2023.121240
Received 27 April 2023; Received in revised form 16 August 2023; Accepted 16 August 2023
Available online 23 August 2023

become shorter with the simultaneous operation of multiple UAVs, numerous tasks can be completed more swiftly. This approach reduces the likelihood of UAVs' batteries being insufficient to sustain their flight over the terrain. The occurrence of UAVs stopping midway through an operation due to diminished energy availability is lessened, thereby reducing the risk of mid-air disruptions.

Similar to any form of robotic swarm, UAV swarms can be utilized in the real world for various activities, just as they would in their individual applications. The primary advantage of the swarm robotics technique lies in its robustness, which manifests in numerous ways. Firstly, a swarm can self-organize or dynamically reorganize how individual robots are deployed, as it comprises many relatively simple agents that are not predetermined for specific roles or duties. Additionally, and for the same reasons, the swarm technique is highly resilient to individual agent failures. There is no singular point of common-mode failure or vulnerability within the swarm due to the entirely decentralized control. In contrast to the substantial technical investment required for achieving fault tolerance in traditional robotic systems, one might argue that the elevated level of robustness observed in UAV swarms is inherent to the swarm robotics methodology (Sahin & Winfield, 2008).

The number of UAV operators required for the initial flight tests with swarms was equivalent to the number of UAVs, significantly increasing the operational costs when employed in groups. Recent advancements have been made in the development of algorithms (Zhao, Zheng, & Liu, 2018) and communications (Campion, Ranganathan, & Faruque, 2018) that allow for the control of the entire swarm by just one person operating the systems. These advancements facilitate more efficient and rapid communication among UAVs, along with improved calculations for collision avoidance paths. This reduces the requirement for human intervention in hazardous situations. Consequently, the latest approaches are geared towards achieving autonomous control of the entire swarm. Flight paths need to be computed at minimal cost while maximizing efficiency. This is known as the Path Planning Problem (Aggarwal & Kumar, 2020), in which the aim is to plan the sequence of movements of robots such as UAVs. Given the often low altitude of their operations, UAVs must navigate around obstacles within the flight area. Consequently, flight path calculations must account for these obstacles and the anticipated positions of all swarm UAVs, in order to prevent collisions among fleet members. The objective is to devise paths that are optimized while circumventing obstacles and other UAVs.

To deal with the complexity of this kind of development, different algorithms are offered in Swarm Intelligence (SI) (Kennedy, 2006). These algorithms aim to coordinate a substantial number of agents concurrently. This coordination relies on a collective of individual actors operating in a self-organized and cohesive manner, while adhering to fundamental, common rules (Bonabeau & Meyer, 2001). In essence, each UAV within the swarm acts as an individual actor. Each actor possesses its information, and its behavior is influenced by its information, the system's rules, and the information shared by other actors. This coordinated behavior is aimed at achieving an objective in the most effective manner (Stentz, 1997).

Certain Path Planning algorithms find utility in military applications. Meanwhile, the scope of civilian applications is relatively restricted, primarily encompassing pursuits or goal-oriented tasks, such as mapping routes through urban areas (Puente-Castro, Rivero, Pazos, & Fernandez-Blanco, 2021). Despite the multitude of potential applications, there exists a scarcity of technologies explicitly designed for agricultural and forestry purposes, particularly those aimed at enhancing the efficiency of field prospecting tasks.

The objective of this research is to create a system that addresses the Path Planning problem within 2D grid-based maps featuring static obstacles and varying quantities of UAVs, accomplished through the utilization of Q-Learning techniques bolstered by Artificial Neural Networks (ANN). Consequently, the principal contributions of this study can be outlined as follows:

1. An innovative Q-Learning-based system that can determine the best possible flying path for a UAV swarm to cover as much area as possible during prospecting activities.
2. A system that can estimate the flight path of any number of UAVs on any sized map with varied sets of obstacles with different shapes and without additional map information such as targets or potential fields.
3. A system capable of calculating paths without the need for a subsequent smoothing stage.
4. A statistical analysis of the results of using a single ANN for each UAV against a global ANN for all UAVs under the same conditions.
5. A path optimization criterion for Q-Learning not dependent on aircraft architecture and capabilities.

The structure of this paper is as follows: An overview of the state of the art is provided in Section 2; a description of the inherent aspects for solving Path Planning problems is developed in Section 3; a description of the technical aspects required for the development of the proposed method is presented in Section 4; a summary of the results of the experimental process is provided in Section 5; the conclusions drawn after evaluating the results, and the possible works and studies to derive the problem to be addressed are provided in Section 6.

2. Background

2.1. Path Planning problems

Path Planning problems involve determining geometric paths for vehicles or robots to follow a set of milestones to reach a designed goal (Gasparetto, Boscariol, Lanzutti, & Vidoni, 2015). Different authors have focused on the development of systems to solve these problems for several years (Patle, Pandey, Parhi, Jagadeesh, et al., 2019). All these authors have employed an extensive array of techniques, spanning from conventional methodologies to Artificial Intelligence approaches (Karur, Sharma, Dharmatti, & Siegel, 2021).

Kong, Nie, and Xu (2022) have put forth a Genetic Algorithm (GA) for controlling swarms within 3D environments. This algorithm underwent testing in a simulator, with results indicating its capability to evade convergence to local maxima. However, it is noteworthy that this approach may entail a relatively higher computational expense in comparison to alternative methods. Liu (2022) have proposed another GA for 3D environments with terrain obstacles. Their method is proficient in deriving smoothed paths without necessitating a subsequent smoothing phase. Additionally, GA can serve as a complementary tool to other algorithms. In their research, they present a system wherein the fitness function is predicated on the UAVs' distance to the final target. This metric, though simplistic, may lead to sluggish approaches if UAVs adopt a spiral trajectory, consequently incurring substantial battery consumption during gradual approaches. In general, the flight environment strongly influences the behavior of the algorithm. That is, a 3D map implies controlling the height of the aircraft while a 2D grid-map implies knowing the state of each cell. To conclude with these techniques, there is a branch within Evolutionary Computation (EC) known as Swarm Intelligence (SI) (Kennedy, 2006) that seeks to mimic the collective behavior of natural systems. For example, Xu, Li, Zhou, Mao, and Huang (2022) have introduced the utilization of GA to optimize a system grounded in the Wolf Pack Algorithm, a purely SI technique, for the coordination of multiple UAVs. Their research illustrates the effectiveness and efficiency of their approach in contrast to alternative methods. However, it is worth noting that they have not presented specific examples of the testing environments for these systems.

Among the category of pure SI techniques, a significant degree of diversity exists. While not as extensively recognized as the methods mentioned earlier, SI techniques have showcased their prowess in optimizing a wide array of problems of diverse natures. This is attributed to their bio-inspired collective behaviors (Minh, Sang-To, Theraulaz, Wahab, & Cuong-Le, 2023; Sang-To, Le-Minh, Mirjalili, Wahab, & Cuong-Le, 2022; Sang-To, Le-Minh, Wahab, & Thanh, 2023). For example, Yang, Zhang, Zhang, and Xiangmin (2019) have harnessed Particle Swarm Optimization (PSO) in conjunction with a voting mechanism to manage multi-UAV control. They have devised an intricately structured voting system tailored to the conventional PSO method, further refining its spatial aspects. Moreover, their innovative approach incorporates time considerations, effectively generating collision-free routes for multiple UAVs within a comparable timeframe. Similarly, Pamosoaji, Piao, and Hong (2019) have employed this algorithm for UAV control. They have carefully factored in the constraints associated with slower aircraft to minimize their flight duration. In their published work, they have demonstrated the algorithm's proficiency in deriving flight paths. However, it is worth noting that they have not provided a quantifiable assessment of the effectiveness or satisfaction of these generated paths. Jain, Yadav, Prakash, Shukla, and Tiwari (2019) put forward the utilization of the Multiverse Optimizer algorithm (MVO) to govern the behavior of multiple UAVs and juxtapose it with the application of a single UAV. This showcases the system's ability to generalize across scenarios. While the system boasts considerable capabilities, it is noteworthy that significant environmental factors impacting aerial operations are not factored into the approach. In a broader sense, one of the predominant limitations of SI techniques is their inclination to converge towards local optima (Yang, 2014). In addition, describing the collective behavior of natural systems is very difficult; it may not be realistic.

2.2. Reinforcement Learning and Q-Learning in Path Planning

Another of the most commonly used set of techniques is Reinforcement Learning (RL) techniques (Puente-Castro, Rivero, et al., 2021). An example of these Artificial Intelligence (AI) techniques can be seen in the research of Qiu, Xu, Wang, Yang, and Liao (2022) where they have implemented an Actor–Critic Reinforcement Learning algorithm to achieve concurrent control of multiple UAVs. Notably, each UAV exclusively possesses local information concerning the environment. This implies that each UAV solely retains its data and does not communicate any information with the other members of the group. Consequently, certain environmental details might be overlooked, or alternatively, some information could potentially be redundantly captured. Additionally, Wei, Huang, et al. (2022) have employed the Actor–Critic RL technique for collaborative data collection across expansive regions. They have introduced a method for estimating energy consumption solely based on time, although a limitation arises from not accounting for the specific type of movement. Consequently, flights involving frequent changes in direction might consume more energy compared to linear flights. To circumvent the challenge of sparse rewards, a common issue in such scenarios, they have implemented an incentive mechanism. Incentives are also applied by Salimi and Pasquier (2021) for the control of a type of UAV group called flocks instead of swarms. In their paper, the authors mention utilizing environments featuring up to 50 obstacles, but unfortunately, they have not provided visual examples. Furthermore, it appears that they depend on the progression of rewards to gauge the system's functionality. However, this approach does not ensure optimal objective completion, as the system might become trapped in a local optimum. A more comprehensive assessment of the system's overall performance, including global-level results, would be necessary to gain deeper insights into its effectiveness.

Continuing with RL techniques, Chen, Dong, Shang, Wu, and Wang (2022) This approach offers a significant advantage, as simulated environments can incorporate intricate details and facilitate the eventual transition to real-world applications. However, it is important to acknowledge a limitation outlined in the paper. Specifically, their focus on cooperative environments. Such environments presuppose that all UAVs will collectively pursue a singular target simultaneously. While this simplifies certain aspects, it does constrain movement flexibility and neglects potential UAV failures or deviations from the uniform path. The use of simulated real environments is also considered by Tu and Juang (2023). In their paper, they utilize the widely adopted AirSim simulator to evaluate the performance of their RL-based system. However, they highlight a limitation in their approach: their system exclusively relies on ultrasonic sensors for obstacle avoidance. Consequently, their UAVs might not effectively detect obstacles constructed from sound-absorbing materials.

A very popular algorithm within RL is Q-Learning (Watkins & Dayan, 1992) and many authors have applied it. This algorithm searches greedily for the best action in each state based on a value given to each available action. By selecting actions with the highest assigned values, it assembles the most optimal sequence of moves. Therefore, it is imperative to accurately determine how to calculate the value attributed to each action. Souto, Alfaia, Cardoso, Araújo, and Francês (2023) have created a system based on Q-Learning, wherein they incorporate external variables unrelated to the UAVs to minimize energy consumption while computing UAV paths. They validate their system's efficacy through simulations conducted within realistic urban environments. The high level of realism exhibited by the simulated environment renders the system readily adaptable to real-world scenarios. Also, de Carvalho et al. (2022) focus on reducing energy consumption by applying Q-Learning techniques. One limitation of their approach is the absence of a defined metric for calculating this consumption. Instead, they have implemented a reward prioritization mechanism based on the type of turn executed by the UAV. It is worth noting that they only account for four specific types of turns based on their angles. Consequently, turns not encompassed within their prioritization framework are not taken into consideration.

2.3. Artificial Neural Networks in Path Planning

Still within AI, a widely used model is the Artificial Neural Network (ANN) (Rosenblatt, 1958). These models are based on Artificial Neurons and have demonstrated their ability to generalize knowledge (McCulloch & Pitts, 1943). Typically, these models are employed to enhance the computation of various essential factors required for path planning, as they can encapsulate a greater depth of knowledge compared to pre-defined formulas. An example of this is the paper by Shiri, Park, and Bennis (2020) where they use ANN to approximate the Hamilton–Jacobi–Bellman equation. Accordingly, the developed ANNs or algorithms reduce biases and can overcome the limitations imposed by the equation. Furthermore, this approach has facilitated the incorporation of wind dynamics into the system, enhancing its reliability and applicability in real-world environments. Another approach is that of Sanna, Godio, and Guglieri (2021), where they use ANN to obtain the best actions to be performed by UAVs. In this manner, they illustrate the system's capacity to acquire additional knowledge by contrasting it with both a non-parametric model and a conventional search model. A similar approach is that of Liu, Zheng, Qin, Zhang, and Yao (2022). Interestingly, they also juxtapose their approach with a classical search algorithm. In contrast to the earlier mentioned paper, the primary objective here is not to cover an entire map. Instead, the focus centers on computing a path between a source point and a destination point. It would indeed be intriguing to comprehend how their system performs in the absence of any indicators, such as those source and destination points. This versatility of not needing points to control the path is also regarded by de Castro et al. (2023). Furthermore, they put forth an alternative approach involving ANN, where it is trained to approximate a conventional search algorithm. This training process involves utilizing the output of the aforementioned algorithm. This innovative strategy allows them to combine the efficiency of a classical search algorithm with the real-time adaptive capabilities inherent in an ANN.

2.4. Artificial Neural Networks applied to Q-Learning in Path Planning

Although the techniques and models described above have demonstrated their capabilities in Path Planning problems, their strength is when used in combination. This is known as Deep Reinforcement Learning (DRL) and consists of training ANNs using RL techniques for Path Planning problems, which is a great advantage because it allows faster abstraction of knowledge in complex environments (Li, 2023).

The Deep Q-Learning or Deep Q-Network (DQN) (Mnih et al., 2015) is one of the most important DRL techniques (Clifton & Laber, 2020). In this case, ANNs are used to enhance the capabilities offered by Q-Learning for Path Planning. For example, Puente-Castro, Rivero, Pazos, and Fernandez-Blanco (2022) propose a dense two-layer ANN applying Q-Learning. The model they introduce is exclusively tested on obstacle-free maps, potentially presenting significant constraints when applied to maps with obstacles. The demonstrated operation is solely time-based, which means its performance would be contingent upon the hardware capabilities of the requisite equipment. Consequently, equipment with superior capabilities would yield improved times, but this would correspondingly entail higher costs. Dhuheir, Baccour, Erbad, Al-Obaidi, and Hamdi (2022) also propose an ANN for their system where they segment a map for each UAV to collect information taking into account latency constraints. While it is important to control such latencies, the test computer is a Rapsberry Pi which a very specific and limited model that, at the date of publication of the article, already has more recent and powerful versions. This is a major limitation because they employ convolutional ANNs, which are known for their high computational cost (Li, Liu, Yang, Peng, & Zhou, 2021). In the paper of Khalil and Rahman (2022) they try to go one step further by making a Federated Learning scheme with an Aggregator (Rieke, Hancox, Li, Milletari, Roth, Albarqouni, Bakas, Galtier, Landman, Maier-Hein, & et al., 2020) applied to a global ANN to converge earlier than those cases where the ANN is trained in the traditional way. Thus, the network acquires more variety of data in less time. The Aggregator module brings together the experience of the ANNs of each UAV that is retrieve individually by each UAV. Therefore, they can train UAVs that escape from hostile systems in the military domain. Within the topic of UAVs in hostile environments, Zhang, Zong, Zhang, Dou, and Tian (2022) propose a similar system but not based on Federated Learning. An added advantage over the previous work is that they test their system in a simulation depicting a complex urban environment, so that the capability of their system can be better seen. There are also publications that make use of DQN but with static targets. An example is the paper by Zhou, Liu, Li, Xu, and Shen (2021), where they address the planning of UAVs swarms with targets. This facilitates the path calculation, but increases the probability that the paths become too dependent on the targets. Similar is the case of Kong, Wang, Gao, and Yu (2023) where they have to establish an allowance threshold error in order to overcome these limitations. Therefore, they have used an ANN that is not only able to calculate the Q-values, but also the distribution of the movements taken by UAVs. Another example of ANN applied to Q-Learning is the work of Raja, Anbalagan, Narayanan, Jayaram, and Ganapathisubramaniyan (2019). Despite not presenting the findings, their paper claims that their technology is scalable to 100 UAVs. In addition to this, a generic system with significant commercial potential can be created by having a system that is scalable to any number of UAVs.

2.5. Summary and contributions

In summary, several papers in the state-of-the-art present a subsequent path-smoothing stage, such as Liu (2022), Susanto et al. (2021) or Correl (2016). By establishing this later step, sharp turns in the paths are modified to make them softer and more gradual. In this way, paths with softer curves are obtained, which lengthens the battery time. However, this process entails increased computational demands, and if the original path contained errors, they could potentially be propagated. Notably, the proposed system omits the path smoothing stage to minimize computation time and preserve the integrity of subsequent data collection, ensuring comprehensive coverage of the terrain during flight.

The authors in the state-of-the-art test their systems on different maps with obstacles but mostly with the same number of UAVs for that given map, like in the work of Kong et al. (2022). In contrast to their suggested model, the proposed system undergoes testing with varying numbers of UAVs to assess its adaptability across different group sizes. Furthermore, maps featuring obstacles of diverse shapes are introduced to ascertain that the system does not merely memorize the obstacle topologies.

Several models in the state-of-the-art require guide points, which can take the form of targets, potential fields, or other indicators. The utilization of these points necessitates the extraction or addition of information to the maps. The dynamic alteration of maps by incorporating new information carries the risk of introducing errors, which could consequently impact the generated paths. Therefore, it is necessary for the map representations chosen to be as complete as possible without the need to add information to them. In the proposed model, maps are tested where only the location of obstacles is indicated and no other information is added.

The main optimization criterion in every work is energy saving. Despite being a common purpose, there are different ways to determine consumption. They are all based on a criterion where an estimation of how much each vehicle can consume according to each type of movement is carried out but these are not equally expensive in different types of UAVs. This highlights the need for a versatile and comprehensive metric, such as the count of movements executed by each UAV. Consequently, a path is considered superior if it entails fewer movements. Moreover, while precise quantitative energy expenditure may remain elusive, it can be inferred that fewer movements inherently translate to reduced energy consumption.

3. Problem formulation

The main aim of this research is to develop a system capable of solving the Path Planning Problem for UAV swarms in maps with obstacles. In scenarios involving multiple vehicles like UAVs, successful Path Planning necessitates the consideration of various variables to ensure optimal efficiency, effectiveness, control, collaboration, and safety. Therefore, it becomes imperative to tackle the challenges posed by these variables, as they form integral components of the overarching goal.

This way of looking at a Path Planning problem as the union of different inherent problems is common in the literature (He, Qi, & Liu, 2021; Puente-Castro et al., 2022). Accordingly, the experimentation process is more precise and organized. The formulation of the Path Planning problem presented is divided into the following areas:

Flight Environments Set
UAV movements
Proposed Model Design
Model Optimization
Model Evaluation Metric

The main point to keep in mind is that solving some aspects of Path Planning problems involves employing simplifications of environment, movement, and other variables simplifications (Giesbrecht, 2004). In the real world, UAVs fly in complex continuous environments. These environments consist of an infinite number of points, and determining the optimal flight path involves exploring infinite combinations. To manage this complexity, the utilization of cell-based maps is a prevalent approach within the field. By dividing the map into finite cells, the exploration process involves fewer combinations, simplifying the task.

However, there is the limitation that the representation of the terrain is greatly simplified, so details that may be crucial for the paths to be calculated may be lost. Even setting up maps divided into cells can complicate the calculation of optimization criteria such as path length because it oversimplifies the representation of the real environment.

It is also necessary to take into account the movements of UAVs. Currently, these aircraft have great flight stability, but their movements are complex and result from the combination of other simpler movements (Susanto et al., 2021). Indeed, to address the challenges related to field coverage and to expedite path calculations, UAV movements are often simplified by being treated as atomic actions without accounting for curves or changes in altitude. Consequently, it is easier to determine whether or not a movement implies that a UAV flies over a cell. On the other hand, the major limitation is to involve abrupt changes in the path, so it may be the case that smooth curves are a better path.

A final limitation to take into account is the tendency of the proposed method in this paper to converge to local maxima by its nature (Jaakkola, Singh, & Jordan, 1994). Therefore, a situation may arise where a good solution is found but a better one is not found. Taking this into account, it is necessary to apply alternatives to reduce this risk.

4. Proposed method

4.1. Reinforcement learning

The solution for the Path Planning Problem for UAVs has been developed by applying Reinforcement Learning (RL) (Sutton & Barto, 2018). Similar to other computational techniques, this method eliminates the necessity of explicitly defining the desired behavior within the system. Instead, a specific component of the algorithm, referred to as an agent, acquires the intended behavior through a process of trial and error. This learning process unfolds within an interactive and dynamic environment, where the agent conducts various tests to adapt and internalize the optimal behavior (Kaelbling, Littman, & Moore, 1996; Wiering & Van Otterlo, 2012). The agent must exploit what it already knows in order to profit from rewards, but it must also explore in order to choose its future actions more wisely. The problem is that pursuing either exploration or exploitation solely would result in failure. The agent must test several different options and gradually favor the ones that seem to work the best. For each action on a stochastic task to gain a valid estimate of the expected reward, several trials must be made. In essence, achieving the ideal behavior requires a dynamic interplay between learning from past experiences and venturing into uncharted territories. All while considering the potential repercussions of the agent's actions on its surroundings.

The explicit consideration of the entire issue of a goal-directed agent interacting with an unpredictable environment is another important aspect of RL. Contrary to many techniques, RL does not take into account sub-problems without considering how they might relate to a bigger one. In other words, it addresses the problem "as a whole".

The learning method differs only slightly in most RL algorithms (Sutton & Barto, 2018). These strategies come in a variety of forms that let the systems handle a wide range of problems. It has been decided to employ a technique known as Q-Learning for this more appropriate to use this study (Watkins & Dayan, 1992). The biggest factor is that, in contrast to other variants, it does not require a model of the environment (model-free approach).

4.1.1. Q-Learning

The agents have to find and follow strategies that allow them to solve problems. These strategies are known as policies. The agents can use their experience to learn the values of all the policies in parallel even when they can only follow one policy at a time thanks to the traditional Q-Learning algorithms (Watkins & Dayan, 1992). The agent learns to follow a policy only through trial and error in this model-free approach (Gläscher, Daw, Dayan, & O'Doherty, 2010). The remarkable convergence property of Q-Learning, known as "greedy convergence", leads to the attainment of an optimal solution regardless of the decision-making policy. This characteristic classifies Q-Learning as an off-policy algorithm. In other words, it only bases its decisions on the agent's interactions with the environment around it. This design ensures the system's adaptability across diverse environments, eliminating the requirement to identify the best policy for each specific scenario. The "Q" in Q-learning stands for "quality" and endeavors to quantify the usefulness of a given action in procuring future rewards.

The most well-known benefit of Q-Learning over other RL techniques is that it allows for the comparison of predicted utility across different actions without the need for an environment model. That means the key factor that led to its selection for this study is how easily it learns and infers situations without requiring their modeling. These algorithms stand out from other RL approaches due to their fundamental distinction: they make decisions based on values stored within a table. These values are known as Q-values, and the table is referred to as a Q-table. The Q-values essentially represent the anticipated reward of an action within the specific context of the environment. From these values, the action with the highest value for each state is chosen. Typically, Bellman's equation (Eq. (1)) is combined with the system's prior predictions to train it. The equation has different elements: $Q(s,a)$ is the function that calculates the Q-value for the current state (s), of the set of states S, and for the given action (a), of the set of actions A, r is the reward of the action taken in that state and it is computed by the reward function $R(s,a)$, γ is the discount factor and $\arg\max_{a'}(Q(s',a'))$ is the maximum computed Q-value of the pair (s',a') represented as $Q(s',a')$. The pair (s',a') is a potential next state–action pair. (s' is the next state and it is given by the transition function $T(s,a)$ which returns the state resulting from the execution of the selected action. The a' is each one of the available actions.

$$Q(s,a) \leftarrow r + \gamma \times \max_{a'}(Q(s',a')) \tag{1}$$

With probability ϵ, a portion of the actions in a Q-Learning problem are made at random, and with probability $1-\epsilon$, the action with the greatest Q-value for that state is adopted. An episode is the series of actions that an agent performs for a certain ϵ until it achieves an end condition (task completion, end of time, etc.) (Shang & Li, 2022). The operation is started over at the beginning of each episode. During testing, episodes reduce the value of ϵ by a factor of reduction. In this approach, the decision of what to do is influenced more by the computed Q-values and less by chance. By considering the minimal value of ϵ, it is kept from becoming too close to zero and to avoid overfitting (Zhang, Vinyals, Munos, & Bengio, 2018).

The key to convergence in Q-Learning is that it is a variant of a Markov Decision Process (MDP). This process is artificially controlled and is known as the action-replay process (ARP) (Watkins & Dayan, 1992). It should be noted that this description assumes a representation of a look-up table, indicating that Q-Learning might not converge correctly for other representations. The requirement that an unlimited number of episodes for each beginning state and action must be included is the most significant implicit condition in the convergence.

Recently, a variant known as Deep Q-learning (DQN) (Mnih et al., 2015) has emerged as an alternative. This approach varies from traditional Q-Learning in that it aims to enhance the calculation of the Q table using Machine Learning (Michie, Spiegelhalter, Taylor, et al., 1994) or Deep Learning (LeCun, Bengio, & Hinton, 2015). The model may deduce the values of the Q table by abstracting sufficient knowledge. In some cases, Bellman's Equation bias concerns can be resolved in this way (Fan, Wang, Xie, & Yang, 2020).

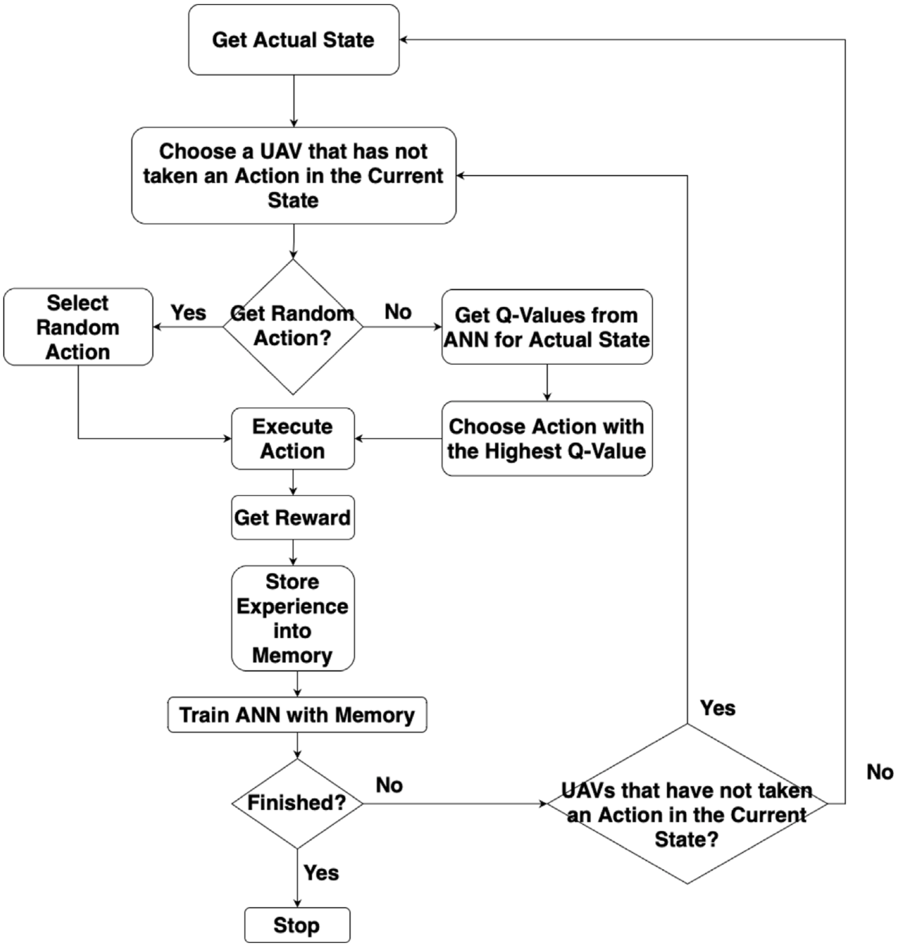

Get Actual State
Choose a UAV that has not taken an Action in the Current State
Get Random Action?
Yes
No
Select Random Action
Get Q-Values from ANN for Actual State
Execute Action
Choose Action with the Highest Q-Value
Get Reward
Store Experience into Memory
Train ANN with Memory
Finished?
No
Yes
UAVs that have not taken an Action in the Current State?
Yes
No
Stop

Fig. 1. Flowchart showing the steps that are followed within each episode of the proposed model. It shows how the ANN interacts with Q-Learning. That is, the ANN learns from the experience gained from performing actions on one or more UAVs.

4.1.2. *Artificial Neural Network*

One of the key points of this paper is the use of ANNs to enhance Q-learning by approximating the Bellman's equation (Krogh, 2008). The authors of this work choose a two-layer fully connected ANN. Unlike convolutional deep ANNs (Albawi, Mohammed, & Al-Zawi, 2017) that other authors have suggested in their studies, it is not assumed that the neighborhood of a cell provides additional information. Hence, it does justify the need to use dense layers (Huang, Liu, Van Der Maaten, & Weinberger, 2017). The only input is the combination of the original environment map, the map with the position of each UAV, and the map with the visited cells and the output are the Q-values for all possible movements. As a result, each Q-Learning experiment follows:

1. Build the ANN model(s) using the selected configuration.
2. Determine the Q-table values and the optimum course of action for each UAV in the swarm using ANN model(s).
3. Train the model(s) based on each chosen action's outcomes.
4. Select the cases where the number of movements required to explore the entire map is lower.

The information available to each agent or UAV in the swarm is very important. If the information is only local (the one perceived by the UAV itself), it implies the loss of information from other UAVs, which can be very useful. Local information may lead to the loss of valuable insights from other UAVs, while global information necessitates efficient communication mechanisms to maintain accurate knowledge updates. Therefore, errors in path planning are reduced. According to previous studies on the state of the art, the system might be employed in two different ways without clear benefits for any of them. The first step is to create a single ANN that will be used to control all of the UAVs moving, determining the movement of each one at each time and verifying the reward received (Fig. 1). Consequently, if a global ANN approach is chosen, all UAVs will share the same design and weights, with their behavior determined solely by their present state. Conversely, adopting a local ANN strategy grants each UAV a distinct ANN, resulting in responses influenced by individual design, weights, and state. That is, the main objective of the experiments is to determine which ANN configuration is better as a controller with respect to the UAVs: one ANN for all UAVs (global ANN), or one ANN for each UAV (local ANN). In both cases, the input data is the same, the information obtained from all UAVs.

4.1.3. *Rewards*

The Reward Function serves as a guiding mechanism within RL problems, providing agents with a framework of rewards and penalties to discern favorable and unfavorable actions. Agents seek to maximize overall gains, i.e. the summation of all rewards in the episode, even at the expense of current actions.

The largest reward must be given in order for the UAV to move to previously unexplored locations. It is also crucial that it grows as fewer cells remain undiscovered (Eq. (2)). In other words, it follows a Hill-Climbing scheme (Kimura, Yamamura, & Kobayashi, 1995). For previously visited cells, another reward is needed. The UAV has a reward in the event that flying over a cell that has previously been visited in order to reach an unvisited one is preferable to flying around it (for example, when there are spurious cells left unvisited). They are given the lowest incentive to prevent UAVs from flying into cells that they are unable to visit. In these situations, the incentive is the lowest and the goal is to maximize rewards. Consequently, UAVs learn that it is best to avoid these situations and opt for the ones that offer higher rewards, which will allow them to maximize the total reward outcome.

$$\text{new cell reward} = \text{new cell base reward} \times (1 + \frac{max(\text{rows}, \text{columns})}{\text{non visited cells}}) \quad (2)$$

4.1.4. Memory Replay

The Memory Replay technique is a prevalent method employed in much of the current research to enhance agents' learning from their interactions with the environment. In this approach, the model undergoes training using a stored set of past observations. These observations encompass a range of information, encompassing the actions taken by the agent as well as the corresponding rewards received. This technique leverages past experiences to enrich the learning process, aiding agents in better understanding and adapting to their environment. Regularly reusing experiences increases sample efficiency and helps in stabilizing the model's training process (Foerster et al., 2017). The memory is designed to retain a substantial number of recent observations, although its capacity is constrained to make optimal use of computational resources. To manage this limitation, the memory employs a First-In-First-Out (FIFO) approach, discarding older observations to accommodate new ones. The memory is capped at a maximum capacity of 60 elements, ensuring a balance between retaining valuable recent experiences and efficiently managing available resources.

In some works in the state-of-the-art, each UAV in the group has a separate memory when using the Memory Replay approach, such as in the paper of Omoniwa, Galkin, and Dusparic (2022). It records observations together with the operations the UAV itself has taken in its memory. The actions of other UAVs are never recorded. This keeps the information from becoming cluttered. Given that multiple UAVs may be located at different locations on the map, the fact that an action is erroneous for one UAV does not always mean that it is improper for others. Moreover, by combining the observations of all UAVs, one UAV may discover actions or combinations of actions that can serve other UAVs later on. The end results might be significantly impacted by the memory's size and structure (Liu & Zou, 2018).

4.1.5. Optimization metric

To estimate the goodness of the proposed method, it has been decided to count the number of actions (also known as movements) performed by all UAVs in the system (Eq. (4)). The number of actions performed by a single UAV is the same as the length of its flight path (Eq. (3)). For a flight path, having too many actions implies higher energy consumption and errors. For instance, it is worse than another path with fewer actions and that flies over the same cells.

Some authors in the state-of-the-art opt for smoothing the paths to make them simpler and better according to an optimization criterion (Correl, 2016). A grid-map will produce paths with several abrupt turns, but a sampling-based technique will produce paths that are randomly zigzagged. Implementing an additional algorithm to smooth the path and reduce some of the sharp turns can notably improve outcomes. However, it is worth noting that path smoothing may introduce inaccuracies in data retrieval since not all cells covered during flight may be completely surveyed, potentially affecting precision. This trade-off between path smoothness and data accuracy underscores the complexity of optimizing UAV trajectories.

$$\text{drone}_i \text{ taken actions} = \text{length}(\text{drone}_i \text{ path}) \quad (3)$$

$$\text{Total actions} = \sum_{i=1}^{n} \text{drone}_i \text{ taken actions} \quad (4)$$

As the desire is to lower the energy consumption for each operation in order to shorten the load time between processes, UAVs are considered to stop once the task is completed and are not considered to automatically return to the starting point. Therefore, the energy consumption of flying back is reduced.

4.1.6. Completeness criterion

As in any Path Planning problem, it is necessary to know if the results are correct. In other words, if they meet a completeness criterion (Giesbrecht, 2004). With this criterion, it is possible to quantify whether each solution obtained is better than the others.

This is a project that seeks to maximize the coverage of a field. That is why the best way to determine completeness is to measure how long it takes the UAVs in a swarm to cover an entire map. This methodology aligns with that of other researchers in the field, who similarly evaluate various parameters with the overarching aim of ascertaining whether all regions of a given map have been successfully surveyed. By employing this comprehensive approach, the project aims to effectively measure the effectiveness of the UAV swarm in achieving optimal coverage across the designated area (Albani, Manoni, Arik, Nardi, & Trianni, 2019; Albani, Nardi, & Trianni, 2017; Qu et al., 2022).

Having a completeness measure that works at the same time as an optimization criterion will allow the proposed method to obtain the best possible path. That is, being able to determine the number of moves the UAVs make to complete the task enables to quantify how good a solution is. Moreover, if a solution fails to cover a map because it converges too early, it will be discarded.

4.2. Experimentation system

As the proposed model for the experiments, a system based on Q-Learning techniques that relies on ANN for better results has been chosen. The best ANN architecture and the best parameters for all precise aspects have been sought through a random hyperparameter search (Bergstra & Bengio, 2012). Thus, the best possible combination of parameters to train the ANNs are obtained in order to have the best possible results.

An ANN made up of two dense layers (Heaton, 2008; Huang et al., 2017), one with 1013 neurons and a ReLU activation function (Agarap, 2018), and the other with 4 neurons and a softmax output function (Gao & Pavel, 2017), has been chosen through empirical experimentation (Fig. 2). The Stochastic Gradient Descend (SGD) (Sutskever, Martens, Dahl, & Hinton, 2013) optimizer was selected as the ANN's optimizer. Two hidden layers have been chosen because architectures from one to three hidden layers have been proven to be universal solutions equivalent to a Turing Machine (Wei, Chen, & Ma, 2022). These kinds of networks can approximate any mapping regardless of the required accuracy, which means it might not be necessary to use a path smoothing stage (Heaton, 2008). The network's input includes the initial environment map, the map with visited cells, and the map indicating UAV positions. This means the ANN uses existing environment data without needing extra information. The ANN's outputs are Q-values for actions in a state, adjusted using the softmax function. There are four distinct Q-values for each movement direction: North, East, South, and West.

To meet all the requirements of the Q-Learning issues explained in Section 4.1.1, an epsilon value (ϵ) of 0.49 has been selected through a preliminary testing process as the probability of making actions at random. The factor of reduction for ϵ equals 0.93, in order not to decrease the value too much and the model continues to learn from the exploration. The minimal value for ϵ is 0.05. The value chosen for the discount factor (γ) is 0.83. All values are selected after previous exploratory research.

The reward values for the agents are in Table 1. The approach employs a dual reinforcement scheme, combining positive and negative reinforcement. New cell discovery is rewarded while revisiting a cell is penalized. This encourages agents to explore new areas rather than revisiting known ones. In addition, passing through an already visited cell is penalized less than passing through a forbidden area. The main reason behind this behavior is that it may be necessary to have paths that cross each other and that is not a mistake. If the rewards were equal, there is a risk that agents would retrace their steps as it is a

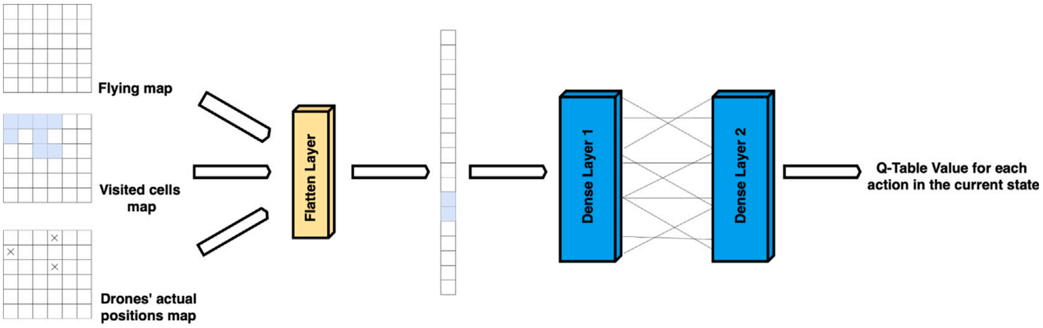

Flying map
Visited cells map
Drones' actual positions map
Flatten Layer
Dense Layer 1
Dense Layer 2
Q-Table Value for each action in the current state

Fig. 2. Diagram with the proposed ANN model. The three inputs are combined into one and the model calculates the Q-values corresponding to each action for the current state like the one proposed in Puente-Castro, Cebrián, Sierra, and Fernandez-Blanco (2021).

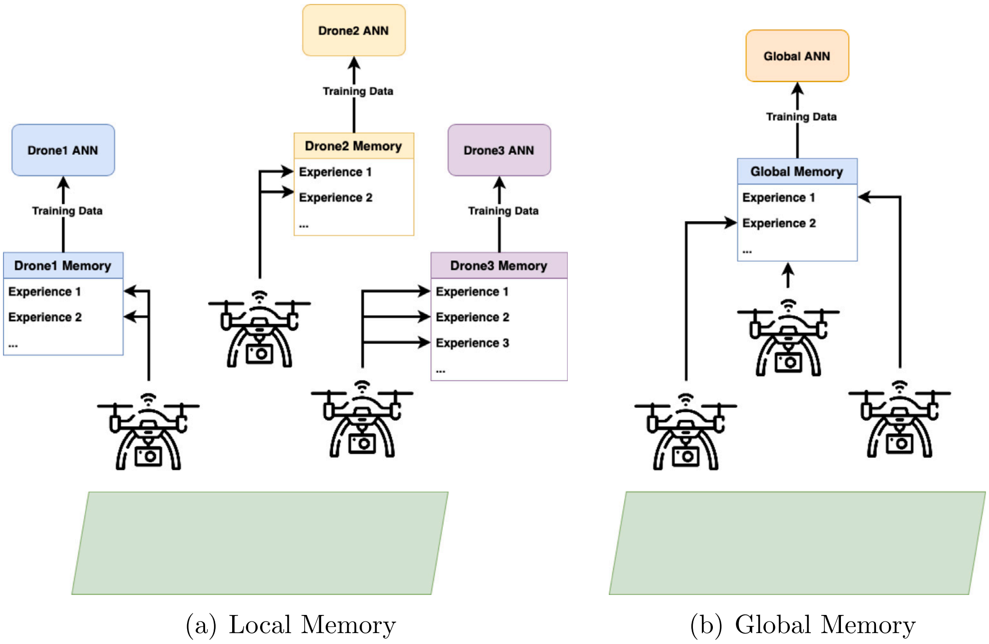


Fig. 3. Diagram illustrating the differences in the UAV experience memory system: Fig. 3(a) shows how UAVs write their experiences (one for each step they take) in their own memories, which will be used to train their own ANNs, so each memory only has experiences from one UAV. Fig. 3(b) illustrates how the UAVs record their experiences in order in a single memory, which will then be used to train a neural network, mixing the experiences of all UAVs together.

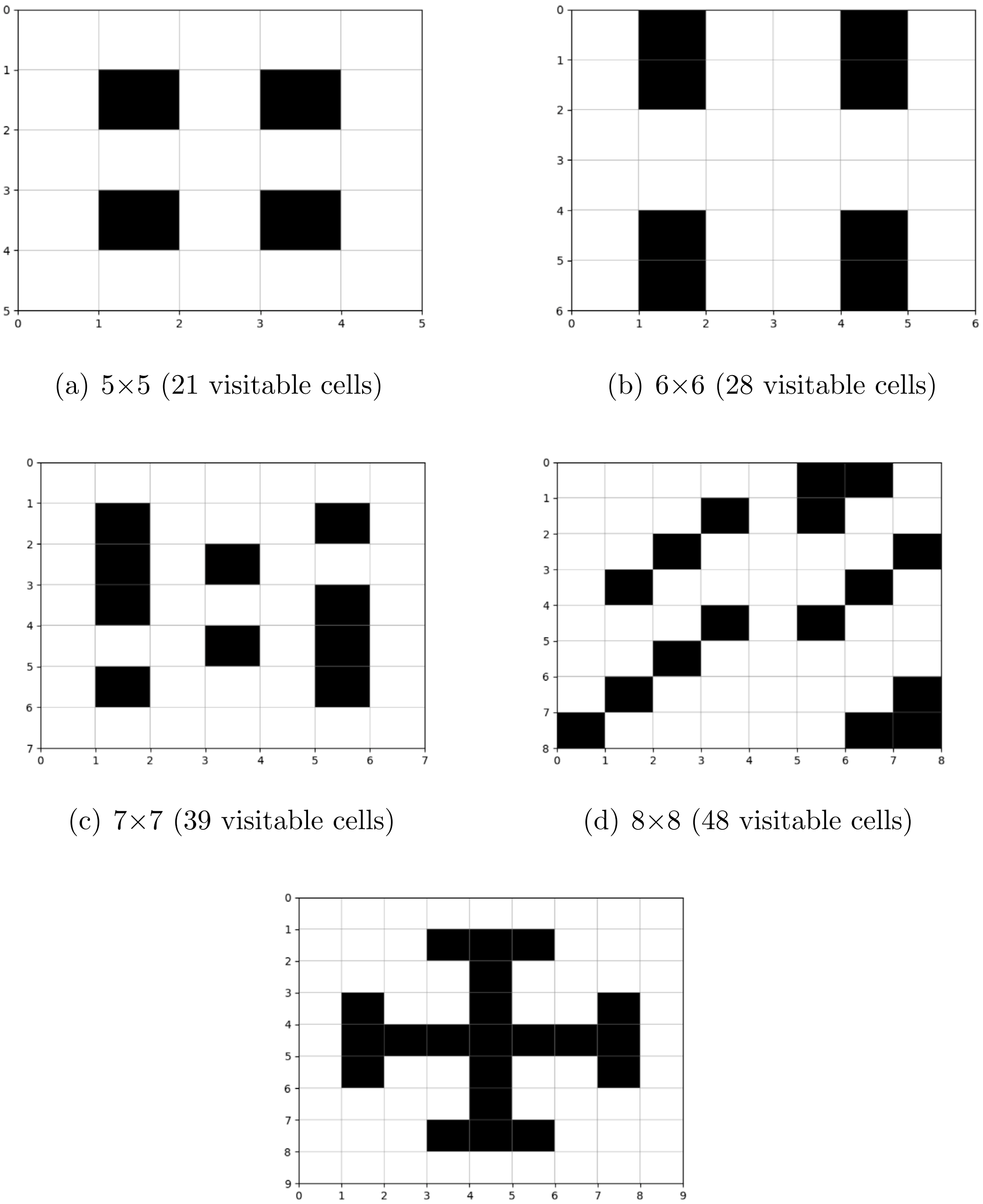
Table 1
Assigned rewards to the various cell types that each UAV visits. The initial rewards values were determined from a prior random exploration (Bergstra & Bengio, 2012) in which the most advantageous reward combinations were chosen.

	Reward
New cell base reward	29.40
Visited cell reward	−31.66
Non-visitable cell	−45.44

reward maximization problem. Therefore, this situation is penalized in case it is not avoidable for the cases in which it is essential to cross paths.

A memory size of 60 actions with their corresponding rewards was selected for this investigation. This choice is driven by the common occurrence of UAV mistakes during the initial phases of the process. A larger memory size is essential to store numerous experiences, considering the total map cells, enabling effective learning from errors. This approach facilitates the avoidance of repeated mistakes and contributes to refining the solution by retraining the model based on the majority of errors. Despite the memory's limitation to 60 elements, it remains vital to assess its behavior in relation to the ANN and its impact on the overall learning process (Fig. 3). Therefore, if it is an ANN per UAV, each ANN will have its memory with the unique experience of a single UAV (Fig. 3(a)). Contrarily, when dealing with a single ANN for all UAVs, it has been decided to use a single collective memory (Fig. 3(b)). Thus, the network learns the cases faced by all UAVs, and, in addition, the data are arbitrarily arranged, similar to having a random buffer in classical Memory Replay (Liu & Zou, 2018). By having the elements arranged randomly, the model is prevented from memorizing movement patterns and learning to generalize flight behavior. In this case, the elements are random but there are elements that represent the experience of each UAV, not just the shuffled experiences of a single UAV.

In addition to the above, the situation in which the system does not find solutions for the given circumstances has been taken into account. Therefore, as a limiting condition for shutdown, the maximum flight time has been set at 30 min. This decision is informed by the typical flight autonomy of commercial UAVs, which often operate for approximately 30 min. Thus, this duration is deemed the upper limit for



(a) 5×5 (21 visitable cells)

(b) 6×6 (28 visitable cells)

(c) 7×7 (39 visitable cells)

(d) 8×8 (48 visitable cells)

(e) 9×9 (60 visitable cells)

Fig. 4. Maps used in the flight environments. Obstacles are shown in black. In white are the cells that can be flown over. UAVs must visit as many white cells as possible.

the UAV swarm's airborne capability, ensuring that the system operates within practical constraints.

To conduct the tests and analyze the results, a set of combinations of map sizes with obstacles and UAV count have been defined. The number of actions carried out by each UAV was taken into consideration when analyzing the results.

4.3. Experiment design

Twenty-five experiments have been created to evaluate the system's capabilities, as presented in this paper. The number of UAVs, the number of ANNs, and the size of the map vary between each one of them.

Since these are ANNs with random initialization, different seeds are tested to have a higher generalization power (Zhang, Ballas, & Pineau, 2018). In addition, for better statistical measurement, the experiments are repeated 5 times with different seeds to have their mean and standard deviation.

Most of the studies in Section 2 employ maps with fixed dimensions ($5 \times 5, 10 \times 10$, or 20×20 cells), but some also use continuous maps without cell division. Continuous maps segmented into uniform cells were chosen over other options for this study. The rationale behind this decision is the paper's focus on complete map coverage for data collection. By employing equally-sized cells, each with identical visitation costs, the objective is to systematically divide the total area into manageable sections. This approach facilitates efficient data collection by ensuring uniform coverage and organized exploration of the entire map.

The selected flight maps have fewer cells compared to the mentioned previous works. This choice is motivated by the consideration of the cost associated with flying over expansive maps. Given that each cell necessitates a stop for surface capture, larger maps would require multiple stops, leading to substantial battery drainage for the UAVs. By employing fewer cells in map division, the frequency of stops and starts for each UAV is reduced, resulting in decreased energy consumption. This strategy aims to optimize energy efficiency during data collection operations. For this purpose, 5 maps have been defined, ranging in size from 5 × 5 cells to 9 × 9 cells (Fig. 4). In the design of the obstacles, the paths were configured in such a way that many changes in direction were compelled and even backward travel was required. This is because they force the UAV to take non-linear paths, which are the ones that have higher energy requirements. In order to establish a common starting point for all maps, the upper right corner has been set. By doing so, it is intended to replicate the fact that operators always begin their operations from a corner.

In the case of the 5 × 5 cell map (Fig. 4(a)), it is intended to simulate the case of tree crops such as olive trees, which are regularly arranged. To complicate that task, the 6 × 6 cell map has been designed (Fig. 4(b)) to display situations that involve turning the UAVs around. In this way, the UAVs are forced to move backward.

Both the 5 × 5 and 6 × 6 cell maps are horizontally and vertically symmetrical. To test how the UAVs behave outside these conditions, the 7 × 7 cell map has been designed (Fig. 4(c)). Furthermore, following this premise, the 8 × 8 cells map has been designed (Fig. 4(d)), which also tests the behavior of the system if the obstacles are arranged diagonally.

The last map to be tested is the 9 × 9 cells map (Fig. 4(e)). In the previous maps, UAVs could pass through the gaps between the obstacles. This map tests the behavior of the system if a single large obstacle has to be circled. In addition, corners have been added to make it more difficult for UAVs to retrace their steps at some points.

Finally, despite varying the obstacles, the number of cells in the maps is also varied to test that the system works for any size. Therefore, the system is tested to prove that it is effective in different situations.

Each chosen map type was evaluated with an increasing number of UAVs because it is crucial for the system to function with any quantity of UAVs. Separate tests with 1, 2, and 3 UAVs have been carried out. Thus, it is demonstrated that the system can adapt to a variety of UAV numbers. It is also worth highlighting the equivalence of employing a global or local approach when a single UAV is used. So, those executions have been referred to as baseline. As a result, it is assumed that the experiment will begin by controlling a single UAV, which is the simplest situation.

In this study, the chosen flight environment has fewer cells compared to the mentioned studies. This decision was influenced by the cost of covering extensive maps, as each cell requires a stop for image capture, leading to high energy consumption. Dividing the map into fewer cells reduces stops and conserves energy, although each cell covers a larger area. Larger captured images offer more contextual information and are better suited for processing, despite having lower detail. Adjusting the cell count to the map size is crucial to prevent loss of information, where a single large cell might miss small obstacles and be treated as an obstacle itself.

The decision to use atomic movements (North, South, East, and West) for the UAVs was made to streamline processing. UAVs can execute these well-defined actions without the need for additional turns. This approach also minimizes energy demands, especially in scenarios with numerous curves that tend to increase energy consumption.

The range of possible movements or actions (a) that UAVs can take was encoded using integer values from 0 to 3, representing the directions: North, East, South, and West. This discrete coding simplifies the representation of movements in the technique and assigns distinct values to each direction.

All these variables are summarized in Table 2.

Table 2
Summary table with the values chosen for experimentation. All the values have been obtained through a preliminary testing process.

Variable	Value
Neurons First Dense Layer	1013 neurons
Activation Function First Dense Layer	ReLU
Neurons Second Dense Layer	4 neurons
Activation Function Second Dense Layer	Linear
ANN Output Function	Softmax
Epsilon (ϵ)	0.49
Epsilon decay	0.93
Minimum Epsilon (ϵ)	0.05
Discount Factor (γ)	0.83
Memory Size	60 actions
Maximum Flight Time	30 min
Maximum Number of Episodes	30 episodes
Possible actions	North, East, South, West

5. Results

Table 3 shows the results obtained from the experimentation. For each map size, the mean and standard deviation of actions taken for each ANN configuration when faced with different numbers of UAVs are compared. To better show the capabilities of the proposed model (known as Proposed in Table 3) it is compared with the model proposed by Puente-Castro et al. (2022), known as Control, which already demonstrated its capabilities on obstacle-free maps. It can be seen that the means of the results of the proposed model are lower than those of the model with which they are contrasted. This can be interpreted as an indication that the paths take fewer actions to complete the operation. Therefore, they are better and more efficient.

With regard to the number of UAVs, it is evident that as the number of UAVs increases, the required number of actions decreases. This supports the notion that coordinated movements among cooperative UAV groups enhance operational speed and efficiency. However, this reduction is not strictly proportional to the number of UAVs, as it is influenced by factors such as map size and obstacles, which vary across different scenarios. For example, the difference in actions required for the 8 × 8 map is greater in all cases than for the 7 × 7 map despite being a map with only 15 more cells.

In the 5 × 5 cell map (Fig. 4(a)) both models present a similar behavior, but the proposed model finds the solution with fewer actions. The local ANNs exhibit higher speed and lower variance compared to global ANNs. The lower variance implies more consistent and optimized paths, showcasing the model's robust behavior. This same pattern is true in the 6 × 6 cell map (Fig. 4(b)). Moreover, in this second map, the obstacles are not islands to go around but form corners that force the UAVs to retrace their steps. Having to retrace their steps is what causes such a large increase in the average movement despite having only 11 more cells, of which 4 are new obstacles.

There is a trend change in the 7 × 7 cell map (Fig. 4(c)), not only because there are more obstacles and the map is larger, but also because the obstacles do not present horizontal or vertical symmetry. In both models, the scenarios involving 2 UAVs exhibit a sudden increase in movement, suggesting potential disruption of paths due to interaction between the UAVs. Interestingly, the proposed global model yielded the lowest mean movement compared to the local model. However, these global paths display higher variance, indicating reduced robustness compared to the local ANN solution.

For the 8 × 8 cell map (Fig. 4(d)) there is no longer such an abrupt growth in the means of the actions of the paths. In this specific scenario, the proposed model does not achieve the lowest mean movement for 2 UAVs. However, it stands out for having the lowest variance, indicating greater accuracy in its computations.

Finally, in the 9 × 9 cell map (Fig. 4(e)) both models behave similarly. It should be noted that the results are similar to those of the

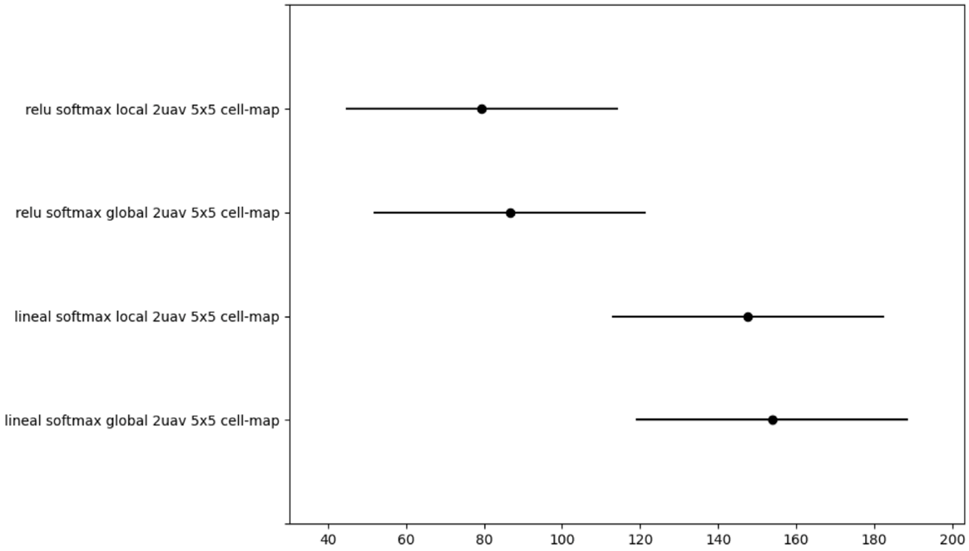


Fig. 5. Plot of the universal confidence interval resulting from Tukey's test. The results for the distributions with statistically significant results are displayed. In the y axis, distributions are listed. In the x axis, the average actions taken for the flight paths of each distribution are displayed.

Table 3
Table with the mean and standard deviation of total actions taken by the swarm of UAVs for each map and for each ANN configuration. Generally, the more UAVs in the swarm, the fewer actions the swarm takes to fly over the entire map.

Map size	Number of UAVs	ANN configuration			
		Local control	Global control	Local proposed	Global proposed
5 × 5	Baseline	283.20 ± 97.79		189.00 ± 91.06	
	2 UAVs	147.60 ± 38.68	153.80 ± 56.42	79.40 ± 7.82	86.60 ± 34.62
	3 UAVs	76.60 ± 53.26	100.60 ± 51.76	56.20 ± 21.32	61.60 ± 47.26
6 × 6	Baseline	503.60 ± 195.34		212.60 ± 49.42	
	2 UAVs	145.40 ± 24.93	232.60 ± 189.49	123.00 ± 12.98	230.00 ± 156.62
	3 UAVs	122.00 ± 55.29	139.60 ± 51.71	127.20 ± 69.83	135.60 ± 51.08
7 × 7	Baseline	523.60 ± 127.93		491.20 ± 15.61	
	2 UAVs	384.80 ± 112.3	537.40 ± 425.41	348.80 ± 151.46	278.40 ± 143.34
	3 UAVs	199.40 ± 66.09	292.20 ± 181.60	166.60 ± 56.00	151.20 ± 72.70
8 × 8	Baseline	1011.00 ± 258.16		1367.80 ± 543.17	
	2 UAVs	700.00 ± 221.08	611.80 ± 484.85	757.60 ± 127.29	654.80 ± 285.69
	3 UAVs	681.80 ± 192.50	582.00 ± 268.86	533.60 ± 369.07	675.8 ± 387.96
9 × 9	Baseline	1332.00 ± 804.16		2264.60 ± 1148.34	
	2 UAVs	980.40 ± 522.45	1107.60 ± 157.47	1232.00 ± 573.74	1087.20 ± 549.05
	3 UAVs	645.00 ± 203.67	690.60 ± 308.33	564.40 ± 210.77	761.00 ± 297.29

8 × 8 map despite being larger, so it can be understood that the layout of the obstacles is more influential to the size of the map.

Statistical tests are performed at a significance level of $\alpha = 0.1$. First, a Shapiro–Wilk (Razali, Wah, et al., 2011) test of normality was performed to find out which statistical significance test can be applied. Not all distributions appear not to follow a normal disposition (Table 4). This phenomenon is more noticeable in scenarios involving multiple UAVs as opposed to a single UAV. The reason behind this could be the interference caused by one UAV's path on the trajectories of others, whether it is due to prior passage through a cell or simultaneous occupancy of the same cell. Essentially, the movement of one UAV has an impact on the paths of both itself and the other UAVs, creating a complex interplay of interactions.

Since not all the distributions obtained do not follow a normal distribution, a Kruskal–Wallis significance test (McKight & Najab, 2010) was used to determine whether they follow significantly different distributions. For this test, a significance level ($alpha$) equal to that used for the normality tests was used.

According to the Kruskal–Wallis test, there are distributions that are significantly different. It is necessary to determine which are significantly different from each other, so a series of Tukey's tests (Tukey, 1949) was performed to find out which are significantly different from each other. The same level of significance was also used for the tests.

Table 4
Table with the p-values of the non-normal distributions resulting from performing the Shapiro–Wilk test (Razali et al., 2011).

Model	Configuration	Number of UAVs	Map size	p-value
Control	Global	2 UAVs	6 × 6	0.095
		3 UAVs	5 × 5	0.019
			7 × 7	0.026
			8 × 8	0.017
	Local	3 UAVs	6 × 6	0.079
			7 × 7	0.075
Proposed	Global	2 UAVs	7 × 7	0.066
			8 × 8	0.094
		3 UAVs	5 × 5	0.011
	Local	2 UAVs	5 × 5	0.049
			8 × 8	0.053
		3 UAVs	6 × 6	0.057

The cases in which there has been statistical significance are those resulting from experimenting with 2 UAVs and maps of 5 × 5. As can be seen in Fig. 5, the indicated distributions show differences if the proposed model is compared with the one used for the contrast (Puente-Castro et al., 2022). These show around half of the average number

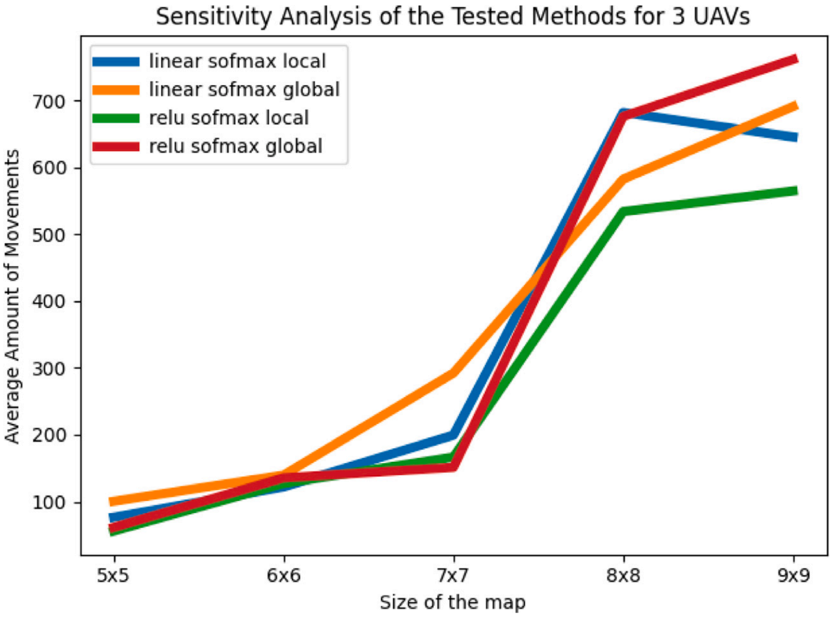


Fig. 6. Sensitivity analysis of the evolution of the models tested under equal conditions. In it, it can be seen that some models have a more stable behavior before smaller maps but that they grow a lot in larger maps. In addition, other models have a less pronounced growth as the size of the maps and the number of obstacles increase.

of actions (x axis) in the proposed model than with the one used to contrast the results. It may be indicative of the models having non-significantly different behavior in all scenarios except for the indicated cases of 2 UAVs on 5 × 5 cells maps. Hence, opting for the proposed model is advantageous due to its tendency to offer shorter or at least equally sized paths. Even though there is no significant difference, these marginal enhancements can prove valuable in practical scenarios. Shorter paths, no matter how minor the difference, contribute to energy savings during flight, making them beneficial in real-world applications.

Both the proposed model and the one involved in the contrast show no significant differences when comparing their global and local variants for the same model. This suggests that the choice between the two approaches might not yield substantial variations in results. Consequently, opting for local models for all UAVs appears favorable, as it typically involves fewer steps and offers comparable outcomes.

The behavior of the contrasted models can be seen in an alternative way by showing a sensitivity analysis between them, as other authors do in the field of RL in economics (Pröllochs, Feuerriegel, & Neumann, 2016). The tested models exhibit structural similarities, yet their diverse parameters and configurations lead to substantial behavioral differences. These variations, while not easily discernible from tabulated data, become evident through sensitivity analysis. By observing how results evolve under different circumstances, we gain insights into the models' behavior. For this purpose, it was decided to look at the evolution of the average number of steps required for 3 UAVs as the complexity of the maps increased (Fig. 6).

6. Conclusions and future work

This study proposes a new system that employs Q-Learning and ANNs with two dense layers to control UAV swarms in maps with obstacles. By optimizing flight paths and reducing actions as the UAV swarm grows, the system offers adaptability across different devices. This shift towards an autonomous UAV swarm provides cost savings, time efficiency, and improved fault tolerance compared to single UAVs or manual management.

Since it is not necessary to know the spatial relationship of the obstacles with the rest of the environment, it can be understood that the sequence of movements and the position of the UAVs in the swarm is more important. Thus, the actions of a single UAV affect the paths of the others, since it modifies the reward values perceived by the others. Additionally, unlike other published work in this field, it is not necessary to include targets or other metrics to guide the computation of paths.

The system has certain limitations. Firstly, the UAV movements are treated atomically, which might not be ideal for tasks needing smoother paths and efficient data capture. The system also does not consider varying UAV heights, potentially affecting path calculations and the accuracy of rewards based on data quality. However, UAVs generally maintain altitudes that accommodate disturbances and adjusting height for obstacles like birds would involve only minor changes. Despite these limitations, the system achieves satisfactory results across different flight heights.

This work provides a basis for further investigation on UAV swarms for Path Planning, particularly concerning experiments with compact fully-connected ANNs in obstacle-ridden maps. Further investigations could encompass more intricate environments like 3D maps, allowing UAVs to execute diverse motions including pitch and roll. Enhancements might involve implementing actions like stopping to mitigate collision risks in intersecting paths.

Enhancing movement precision can entail increased system complexity. For instance, integrating ANNs for distinct functions could be explored. The combination of multiple ANNs offers the potential to incorporate additional flight capabilities, like altitude adjustments or tilting. Employing multiple ANNs to coordinate composite movements, such as simultaneous ascent and turns, may lead to improved accuracy and quicker outcomes.

The most important improvement is to achieve a system that allows a greater variety of movements. For example, these actions can be combinations in different degrees of the above. The "stop" command could even be used as an action. Having more actions and some combined ones makes it more difficult to count the paths, but it can improve the precision of the movements. In this way, the data capture is optimized and the risk of maneuvers is reduced.

Code availability

Source code and a Docker container are available at:

https://github.com/TheMVS/UAV_SWARMS_RL_FIXED_OBSTACLES_MAPS

https://hub.docker.com/repository/docker/themvs/uav_swarms_rl_fixed_obstacles_maps/

Funding

This project was supported by the FCT - Foundation for Science and Technology, Portugal, in the context of the project [grant number UIDB/00127/
2020], and also POCI 2020, in the context of the Germirrad project [grant number POCI-01-0247-FEDER-072237]. Also, the General Directorate of Culture, Education, and University Management of Xunta de Galicia [grant number ED431D 2017/16]. This work was also funded by the grant for the consolidation and structuring of competitive research units [grant number ED431C 2022/46] from the General Directorate of Culture, Education and University Management of Xunta de Galicia, and the CYTED network, Spain [grant number PCI2018_093284] funded by the Spanish Ministry of Innovation and Science. This project was also supported by the General Directorate of Culture, Education and University Management of Xunta de Galicia "PRACTICUM DIRECT" [grant number IN845D-2020/03].

CRediT authorship contribution statement

Alejandro Puente-Castro: Conceptualization, Methodology, Software, Validation, Formal analysis, Resources, Data curation, Writing – original draft, Visualization, Investigation. **Daniel Rivero:** Writing – review & editing, Supervision. **Eurico Pedrosa:** Writing – review & editing. **Artur Pereira:** Writing – review & editing. **Nuno Lau:** Writing – review & editing, Supervision. **Enrique Fernandez-Blanco:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

Agarap, A. F. (2018). Deep learning using rectified linear units (ReLU). arXiv preprint arXiv:1803.08375.

Aggarwal, S., & Kumar, N. (2020). Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges. *Computer Communications*, *149*, 270–299.

Albani, D., IJsselmuiden, J., Haken, R., & Trianni, V. (2017). Monitoring and mapping with robot swarms for agricultural applications. In *2017 14th IEEE international conference on advanced video and signal based surveillance* (pp. 1–6). IEEE.

Albani, D., Manoni, T., Arik, A., Nardi, D., & Trianni, V. (2019). Field coverage for weed mapping: Toward experiments with a UAV swarm. In *Bio-inspired information and communication technologies: 11th EAI international conference, BICT 2019, Pittsburgh, PA, USA, March 13–14, 2019, Proceedings 11* (pp. 132–146). Springer.

Albani, D., Nardi, D., & Trianni, V. (2017). Field coverage and weed mapping by UAV swarms. In *2017 IEEE/RSJ international conference on intelligent robots and systems* (pp. 4319–4325). Ieee.

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 International conference on engineering and technology* (pp. 1–6). Ieee.

Austin, R. (2011). *Unmanned aircraft systems: UAVs design, development and deployment*. John Wiley & Sons.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*(2).

Bocchino, R., Canham, T., Watney, G., Reder, L., & Levison, J. (2018). F Prime: An open-source framework for small-scale flight software systems. Preprint.

Bonabeau, E., & Meyer, C. (2001). Swarm intelligence: A whole new way to think about business. *Harvard Bus. Rev.*, *79*(5), 106–115.

Campion, M., Ranganathan, P., & Faruque, S. (2018). A review and future directions of UAV swarm communication architectures. In *2018 IEEE international conference on electro/information technology* (pp. 0903–0908). IEEE.

de Carvalho, K. B., de Oliveira, I. R. L., Villa, D. K., Caldeira, A. G., Sarcinelli-Filho, M., & Brandão, A. S. (2022). Q-learning based path planning method for uavs using priority shifting. In *2022 International conference on unmanned aircraft systems* (pp. 421–426). IEEE.

de Castro, G. G., Pinto, M. F., Biundini, I. Z., Melo, A. G., Marcato, A. L., & Haddad, D. B. (2023). Dynamic path planning based on neural networks for aerial inspection. *Journal of Control, Automation and Electrical Systems*, *34*(1), 85–105.

Chen, Y., Dong, Q., Shang, X., Wu, Z., & Wang, J. (2022). Multi-UAV autonomous path planning in reconnaissance missions considering incomplete information: A reinforcement learning method. *Drones*, *7*(1), 10.

Clifton, J., & Laber, E. (2020). Q-learning: Theory and applications. *Annual Review of Statistics and Its Application*, *7*, 279–301.

Correl, P. (2016). Introduction to autonomous robots. Kinematics, perception, localization and planning. (pp. 85–86).

Corte, A. P. D., Souza, D. V., Rex, F. E., Sanquetta, C. R., Mohan, M., Silva, C. A., et al. (2020). Forest inventory with high-density UAV-lidar: Machine learning approaches for predicting individual tree attributes. *Computers and Electronics in Agriculture*, *179*, Article 105815.

Dhuheir, M., Baccour, E., Erbad, A., Al-Obaidi, S. S., & Hamdi, M. (2022). Deep reinforcement learning for trajectory path planning and distributed inference in resource-constrained UAV swarms. *IEEE Internet of Things Journal*.

Fan, J., Wang, Z., Xie, Y., & Yang, Z. (2020). A theoretical analysis of deep Q-learning. In *Learning for dynamics and control* (pp. 486–489). PMLR.

Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P. H., Kohli, P., et al. (2017). Stabilising experience replay for deep multi-agent reinforcement learning. In *International conference on machine learning* (pp. 1146–1155). PMLR.

Gao, B., & Pavel, L. (2017). On the properties of the softmax function with application in game theory and reinforcement learning. arXiv preprint arXiv:1704.00805.

Gasparetto, A., Boscariol, P., Lanzutti, A., & Vidoni, R. (2015). Path planning and trajectory planning algorithms: A general overview. *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*, 3–27.

Giesbrecht, J. (2004). *Global path planning for unmanned ground vehicles: Technical report*, Defence Research and Development Suffield (ALBERTA).

Gläscher, J., Daw, N., Dayan, P., & O'Doherty, J. P. (2010). States versus rewards: Dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron*, *66*(4), 585–595.

He, W., Qi, X., & Liu, L. (2021). A novel hybrid particle swarm optimization for multi-UAV cooperate path planning. *Applied Intelligence*, *51*(10), 7350–7364.

Heaton, J. (2008). *Introduction to neural networks with Java* (p. 158). Heaton Research, Inc..

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4700–4708).

Huuskonen, J., & Oksanen, T. (2018). Soil sampling with drones and augmented reality in precision agriculture. *Computers and Electronics in Agriculture*, *154*, 25–35.

Jaakkola, T., Singh, S., & Jordan, M. (1994). Reinforcement learning algorithm for partially observable Markov decision problems. *Advances in Neural Information Processing Systems*, *7*.

Jain, G., Yadav, G., Prakash, D., Shukla, A., & Tiwari, R. (2019). MVO-based path planning scheme with coordination of UAVs in 3-D environment. *Journal of Computer Science*, *37*, Article 101016.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.

Karur, K., Sharma, N., Dharmatti, C., & Siegel, J. E. (2021). A survey of path planning algorithms for mobile robots. *Vehicles*, *3*(3), 448–468.

Kennedy, J. (2006). Swarm intelligence. In *Handbook of nature-inspired and innovative computing* (pp. 187–219). Springer.

Khalil, A. A., & Rahman, M. A. (2022). FED-UP: Federated deep reinforcement learning-based UAV path planning against hostile defense system. In *2022 18th international conference on network and service management* (pp. 268–274). IEEE.

Kimura, H., Yamamura, M., & Kobayashi, S. (1995). Reinforcement learning by stochastic hill climbing on discounted reward. In *Machine learning proceedings 1995* (pp. 295–303). Elsevier.

Kong, F., Nie, Y., & Xu, X. (2022). An improved GA-based approach for UAV swarm formation transformation. In *2022 IEEE 6th information technology and mechatronics engineering conference, vol. 6* (pp. 1715–1720). IEEE.

Kong, F., Wang, Q., Gao, S., & Yu, H. (2023). B-APFDQN: A UAV path planning algorithm based on deep Q-network and artificial potential field. *IEEE Access*.

Krogh, A. (2008). What are artificial neural networks? *Nature biotechnology*, *26*(2), 195–197.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Li, S. E. (2023). Deep reinforcement learning. In *Reinforcement learning for sequential decision and optimal control* (pp. 365–402). Springer.

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2021). A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*.

Liu, J. (2022). An improved genetic algorithm for rapid UAV path planning. *Journal of Physics: Conference Series*, *2216*, Article 012035.

Liu, J., Wang, W., Wang, T., Shu, Z., & Li, X. (2018). A motif-based rescue mission planning method for UAV swarms usingan improved PICEA. *IEEE Access*, *6*, 40778–40791.

Liu, Y., Zheng, Z., Qin, F., Zhang, X., & Yao, H. (2022). A residual convolutional neural network based approach for real-time path planning. *Knowledge-Based Systems*, *242*, Article 108400.

Liu, R., & Zou, J. (2018). The effects of memory replay in reinforcement learning. In *2018 56th Annual Allerton conference on communication, control, and computing* (pp. 478–485). IEEE.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*, 115–133.

McKight, P. E., & Najab, J. (2010). Kruskal-Wallis test. In *The corsini encyclopedia of psychology* (p. 1). Wiley Online Library.

Michie, D., Spiegelhalter, D. J., Taylor, C., et al. (1994). Machine learning. *Neural and Statistical Classification*, *13*.

Minh, H. L., Sang-To, T., Theraulaz, G., Wahab, M. A., & Cuong-Le, T. (2023). Termite life cycle optimizer. *Expert Systems with Applications*, *213*, Article 119211.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Noor, N. M., Abdullah, A., & Hashim, M. (2018). Remote sensing UAV/drones and its applications for urban areas: A review. In *IOP Conference Series: Earth and Environmental Science*: *vol. 169*, IOP Publishing, Article 012003.

Omoniwa, B., Galkin, B., & Dusparic, I. (2022). Optimizing energy efficiency in UAV-assisted networks using deep reinforcement learning. *IEEE Wireless Communications Letters*, *11*(8), 1590–1594.

Pamosoaji, A. K., Piao, M., & Hong, K.-S. (2019). PSO-based minimum-time motion planning for multiple vehicles under acceleration and velocity limitations. *International Journal of Control, Automation and Systems*, *17*(10), 2610–2623.

Patle, B., Pandey, A., Parhi, D., Jagadeesh, A., et al. (2019). A review: On path planning strategies for navigation of mobile robot. *Defence Technology*, *15*(4), 582–606.

Pröllochs, N., Feuerriegel, S., & Neumann, D. (2016). Detecting negation scopes for financial news sentiment using reinforcement learning. In *2016 49th Hawaii international conference on system sciences* (pp. 1164–1173). IEEE.

Puente-Castro, A., Cebrián, D., Sierra, A., & Fernandez-Blanco, E. (2021). Artificial intelligence techniques for autonomous drone swarms. In *MOL2NET́21, Conference on molecular, biomedical & computational sciences and engineering* (7th ed.). MDPI.

Puente-Castro, A., Rivero, D., Pazos, A., & Fernandez-Blanco, E. (2021). A review of artificial intelligence applied to path planning in UAV swarms. *Neural Computing and Applications*, 1–18.

Puente-Castro, A., Rivero, D., Pazos, A., & Fernandez-Blanco, E. (2022). UAV swarm path planning with reinforcement learning for field prospecting. *Applied Intelligence*, 1–18.

Qiu, X., Xu, L., Wang, P., Yang, Y., & Liao, Z. (2022). A data-driven packet routing algorithm for an un-manned aerial vehicle swarm: A multi-agent reinforcement learning approach. *IEEE Wireless Communications Letters*.

Qu, C., Boubin, J., Gafurov, D., Zhou, J., Aloysius, N., Nguyen, H., et al. (2022). Uav swarms in smart agriculture: Experiences and opportunities. In *2022 IEEE 18th international conference on E-science* (pp. 148–158). IEEE.

Rabinovitch, J., Lorenz, R., Slimko, E., & Wang, K. S. C. (2021). Scaling sediment mobilization beneath rotorcraft for Titan and Mars. *Aeolian Research*, *48*, Article 100653.

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* (pp. 1089–1095). IEEE.

Razali, N. M., Wah, Y. B., et al. (2011). Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of Statistical Modeling and Analytics*, *2*(1), 21–33.

Rieke, N., Hancox, J., Li, W., Milletari, F., Roth, H., Albarqouni, S., et al. (2020). The future of digital health with federated learning. *NPJ digital medicine*, *3*, 119.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386.

Sahin, E., & Winfield, A. F. (2008). Special issue on swarm robotics. *Swarm Intelligence*, *2*(2–4), 69–72.

Salimi, M., & Pasquier, P. (2021). Deep reinforcement learning for flocking control of UAVs in complex environments. In *2021 6th international conference on robotics and automation engineering* (pp. 344–352). IEEE.

Sang-To, T., Le-Minh, H., Mirjalili, S., Wahab, M. A., & Cuong-Le, T. (2022). A new movement strategy of grey wolf optimizer for optimization problems and structural damage identification. *Advances in Engineering Software*, *173*, Article 103276.

Sang-To, T., Le-Minh, H., Wahab, M. A., & Thanh, C.-L. (2023). A new metaheuristic algorithm: Shrimp and Goby association search algorithm and its application for damage identification in large-scale and complex structures. *Advances in Engineering Software*, *176*, Article 103363.

Sanna, G., Godio, S., & Guglieri, G. (2021). Neural network based algorithm for multi-UAV coverage path planning. In *2021 International conference on unmanned aircraft systems* (pp. 1210–1217). IEEE.

Shang, Y., & Li, S. (2022). Hybrid combinatorial remanufacturing strategy for medical equipment in the pandemic. *Computers & Industrial Engineering*, Article 108811.

Shiri, H., Park, J., & Bennis, M. (2020). Remote UAV online path planning via neural network-based opportunistic control. *IEEE Wireless Communications Letters*, *9*(6), 861–865.

Souto, A., Alfaia, R., Cardoso, E., Araújo, J., & Francês, C. (2023). UAV path planning optimization strategy: Considerations of urban morphology, microclimate, and energy efficiency using Q-learning algorithm. *Drones*, *7*(2), 123.

Stentz, A. (1997). Optimal and efficient path planning for partially known environments. In *Intelligent unmanned ground vehicles* (pp. 203–220). Springer.

Susanto, T., Setiawan, M. B., Jayadi, A., Rossi, F., Hamdhi, A., & Sembiring, J. P. (2021). Application of unmanned aircraft PID control system for roll, pitch and yaw stability on fixed wings. In *2021 International conference on computer science, information technology, and electrical engineering* (pp. 186–190). IEEE.

Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning* (pp. 1139–1147). PMLR.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Tu, G. T., & Juang, J. G. (2023). UAV path planning and obstacle avoidance based on reinforcement learning in 3D environments. In *Actuators*: *vol. 12*, (no. 2), (p. 57). MDPI.

Tukey, J. W. (1949). Comparing individual means in the analysis of variance. *Biometrics*, 99–114.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*(3), 279–292.

Wei, C., Chen, Y., & Ma, T. (2022). Statistically meaningful approximation: A case study on approximating turing machines with transformers. *Advances in Neural Information Processing Systems*, *35*, 12071–12083.

Wei, K., Huang, K., Wu, Y., Li, Z., He, H., Zhang, J., et al. (2022). High-performance UAV crowdsensing: A deep reinforcement learning approach. *IEEE Internet of Things Journal*.

Wiering, M., & Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, learning, and optimization*, *12*, 3.

Xu, S., Li, L., Zhou, Z., Mao, Y., & Huang, J. (2022). A task allocation strategy of the UAV swarm based on multi-discrete wolf pack algorithm. *Applied Sciences*, *12*(3), 1331.

Yang, X. S. (2014). Swarm intelligence based algorithms: A critical analysis. *Evolutionary Intelligence*, *7*(1), 17–28.

Yang, L., Zhang, X., Zhang, Y., & Xiangmin, G. (2019). Collision free 4D path planning for multiple UAVs based on spatial refined voting mechanism and PSO approach. *Chinese Journal of Aeronautics*, *32*(6), 1504–1519.

Yeaman, M. L., & Yeaman, M. (1998). *Virtual air power: A case for complementing ADF air operations with uninhabited aerial vehicles*. Air Power Studies Centre.

Zhang, A., Ballas, N., & Pineau, J. (2018). A dissection of overfitting and generalization in continuous reinforcement learning. arXiv preprint arXiv:1806.07937.

Zhang, C., Vinyals, O., Munos, R., & Bengio, S. (2018). A study on overfitting in deep reinforcement learning. arXiv preprint arXiv:1804.06893.

Zhang, R., Zong, Q., Zhang, X., Dou, L., & Tian, B. (2022). Game of drones: Multi-uav pursuit-evasion game with online motion planning by deep reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhao, Y., Zheng, Z., & Liu, Y. (2018). Survey on computational-intelligence-based UAV path planning. *Knowledge-Based Systems*, *158*, 54–64.

Zhou, W., Liu, Z., Li, J., Xu, X., & Shen, L. (2021). Multi-target tracking for unmanned aerial vehicle swarms using deep reinforcement learning. *Neurocomputing*, *466*, 285–297.

Alejandro Puente-Castro BSc. in Computer Science, gained his MSc. in Bioinformatics for Health Sciences and has worked infields, such as early detection of Alzheimer's disease using Deep Learning techniques or self-quantification. Currently, his research is focused on applying Artificial Intelligence techniques to the coordination of heterogeneous groups of Unmanned Aerial Vehicles (UAVs) and Bioinformatics.