TU Delft

Delft University of Technology

Learning-Based Multi-UAV Flocking Control With Limited Visual Field and Instinctive Repulsion

Bai, Chengchao; Yan, Peng; Piao, Haiyin; Pan, Wei; Guo, Jifeng

DOI
10.1109/TCYB.2023.3246985

Publication date
2024

Document Version
Final published version

Published in
IEEE Transactions on Cybernetics

Citation (APA)
Bai, C., Yan, P., Piao, H., Pan, W., & Guo, J. (2024). Learning-Based Multi-UAV Flocking Control With Limited Visual Field and Instinctive Repulsion. *IEEE Transactions on Cybernetics*, *54*(1), 462-475. https://doi.org/10.1109/TCYB.2023.3246985

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

https://www.openaccess.nl/en/you-share-we-take-care

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

Learning-Based Multi-UAV Flocking Control With Limited Visual Field and Instinctive Repulsion

Chengchao Bai, *Member, IEEE*, Peng Yan, Haiyin Piao, Wei Pan, *Member, IEEE*, and Jifeng Guo, *Member, IEEE*

***Abstract*—This article explores deep reinforcement learning (DRL) for the flocking control of unmanned aerial vehicle (UAV) swarms. The flocking control policy is trained using a centralized-learning-decentralized-execution (CTDE) paradigm, where a centralized critic network augmented with additional information about the entire UAV swarm is utilized to improve learning efficiency. Instead of learning inter-UAV collision avoidance capabilities, a repulsion function is encoded as an inner-UAV "instinct." In addition, the UAVs can obtain the states of other UAVs through onboard sensors in communication-denied environments, and the impact of varying visual fields on flocking control is analyzed. Through extensive simulations, it is shown that the proposed policy with the repulsion function and limited visual field has a success rate of 93.8% in training environments, 85.6% in environments with a high number of UAVs, 91.2% in environments with a high number of obstacles, and 82.2% in environments with dynamic obstacles. Furthermore, the results indicate that the proposed learning-based methods are more suitable than traditional methods in cluttered environments.**

***Index Terms*—Deep reinforcement learning (DRL), flocking control, inter-unmanned aerial vehicle (UAV) collision avoidance, limited visual field, UAVs.**

I. Introduction

RECENTLY, unmanned aerial vehicle (UAV) swarms [1] have attracted increasing attention due to their superior mission efficiency and robustness compared to the deployment of a single UAV, and have a wide range of applications, such as in communication services [2], detection and surveillance [3], reconstruction and mapping [4], and agriculture [5]. The cooperative control of a UAV swarm is essential to ensure the success of a mission. In particular, flocking control is an effective method to control a UAV swarm collaboratively.

Much research has been conducted on the flocking control problem [6]. Some authors have treated the UAV swarm control as a multiobjective optimization problem and have employed optimization algorithms to solve it. For example, in [7] an evolutionary optimization framework was proposed to solve the collective motion problem of aerial robots in confined spaces where motion constraints, communication status, and perturbations were explicitly modeled. In [8], the flocking control of a UAV swarm was formulated as a multiobjective optimization problem, and the multiobjective pigeon-inspired optimization (MPIO) was modified based on the hierarchical learning behavior of pigeon flocks to solve the problem in a distributed manner. The aforementioned methods achieved collision avoidance between UAVs through mutual repulsion. When the distance between UAVs is less than the range of repulsion, the repulsive force pulls the UAVs away in opposite directions, which can cause jitter in the control commands. The algorithm may fall into local minimum traps when the number of UAVs is high. Lyu et al. [9] formulated the multivehicle flocking control problem in a model predictive control (MPC) scheme where the vehicles are driven to follow a commonly desired trajectory, and collision avoidance is considered a necessary condition by setting it as an optimization constraint. However, the limited computing power of the onboard computer can make it challenging for an optimization algorithm to find the optimal result in a limited time. The models of the environment and UAVs are difficult to obtain in practice, which further limits the performance of the optimization algorithm.

Several studies have been conducted based on Reynolds' three heuristic rules of flocking control [10]. Olfati-Saber [11] proposed a theoretical framework for the design and analysis of distributed flocking algorithms and proposed the concepts of α-agents, β-agents, and γ-agents. Liu and Gao [12] further improved the flocking algorithm in [11] by using a virtual leader to ensure the information security of UAV swarms and optimize the communication performance between UAVs. In [13], three modes of control protocol and a state-dependent switching logic were designed to address complex constraints such as nonholonomic constraints, speed limits, and efficient use of airspace. Although the above methods use simple rules to implement flocking control, they assume that UAVs can communicate to obtain the states of other UAVs and do not consider communication denial environments.

Manuscript received 11 August 2022; revised 15 November 2022 and 28 January 2023; accepted 12 February 2023. Date of publication 8 March 2023; date of current version 19 December 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61973101; in part by the Young Elite Scientists Sponsorship Program by CAST under Grant 2021QNRC001; and in part by the Aeronautical Science Foundation of China under Grant 20180577005. This article was recommended by Associate Editor Y. Pan. *(Corresponding authors: Chengchao Bai; Haiyin Piao.)*

Chengchao Bai, Peng Yan, and Jifeng Guo are with the School of Astronautics, Harbin Institute of Technology, Harbin 150001, China (e-mail: baichengchao@hit.edu.cn; yanpeng@hit.edu.cn; guojifeng@hit.edu.cn).

Haiyin Piao is with the School of Electronics and Information, Northwestern Polytechnical University, Xian 710000, China, and also with the AI Center, SADRI Institute, Shenyang 110035, China (e-mail: haiyinpiao@mail.nwpu.edu.cn).

Wei Pan is with the Department of Cognitive Robotics, Delft University of Technology, 2628 CD Delft, The Netherlands, and also with the Department of Computer Science, University of Manchester, M13 9PL Manchester, U.K. (e-mail: wei.pan@manchester.ac.uk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2023.3246985.

Digital Object Identifier 10.1109/TCYB.2023.3246985

2168-2267 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

In the flocking control of the UAV swarm, the inter-UAV information interaction is generally achieved through communication networks [14]. However, the vision-based estimation of the inter-UAV state is a more reliable method in communication denial environments. Still, only a few studies have considered the impact of the visual field of the UAV on the control of the UAV swarm. In [15], a relative positioning solution was proposed for the rendezvous and close formation flight of UAVs based on IR cameras with a nonlinear estimation framework. However, this work did not focus on flocking control of UAVs in complex environments, and the environment used for flight experiments was relatively simple. In [16], a vision-assisted flocking system was proposed, but it only considered the case of a small number of UAVs, and it is not clear how it can be extended to a large-scale UAV swarm. Therefore, there is still a need for further research on the flocking control of UAV swarms in communication denial environments, particularly focusing on the impact of the visual field of the UAV.

Accurate UAV models are crucial for UAV motion control [17]. However, building an accurate UAV model is challenging due to uncertain nonlinearities and the strong coupling of the UAV model, mainly when there are external disturbances. To address this issue, Wang et al. [18] developed a neural observer to estimate the unknown state variables of a quadrotor UAV. Elhaki and Shojaei [19] proposed a model-free controller for quadrotors in a reinforcement learning (RL) framework [20] based on actor-critic neural networks, which only requires measurable signals of the closed-loop system and does not rely on the UAV model. Furthermore, due to the model-free characteristics of RL and the strong representational ability of deep neural networks (DNNs), deep reinforcement learning (DRL) has a wide range of applications in complex multiagent systems [21]. For example, Wang and Chen [22] investigated linear multiagent systems' optimal containment control problem through a model-free approach, where DNN-based Q-functions and control policies were used to improve the proposed approach. In [23], a multiagent actor-critic algorithm was proposed, which can construct an interpretable interaction structure in dynamic environments.

DRL has also been extensively used in UAV flock control. In [24], RL and flocking control were combined to enable a multirobot system to learn collaboratively to avoid predators while maintaining network topology and connectivity. However, the RL-based decision module operates in a discrete space, making it challenging to find the appropriate safe place when encountering complex and dynamic environments. Hung and Givigi [25] proposed a Q-learning-based approach to teach followers how to flock in a leader–follower topology. Based on the work in [25], the leader–follower flocking problem was addressed in [26] in continuous state and action spaces. The authors developed an actor-critic RL approach for flocking control, known as the continuous actor-critic with experience replay (CACER). However, in the studies by both [25] and [26], altitude differences were used to avoid collisions among the UAVs, and collision avoidance among UAVs at the same altitude was not considered.

In addition, there have been some methods to enable UAVs to learn to avoid each other by designing reward functions. Yan et al. [27] addressed the collision-free flocking problem of fixed-wing UAVs in a DRL framework. Specifically, the collision risks of other nearby UAVs were constructed as a fixed-size local situation map. A DRL algorithm was then used to learn the collision-free flocking behavior. Xu et al. [28] used a deep deterministic policy gradient (DDPG) to learn the flocking control policy with collision avoidance and communication preservation. Wang et al. [29] used DDPG to learn a policy that enables UAVs to flock and perform navigation tasks in complex environments where each UAV only considers the relative position of the nearest two neighbors on its left and right sides. Both in [27] and [28], a centralized-training-decentralized-execution (CTDE) [30], [31] paradigm was used to train the control policies. However, the methods of [27], [28], and [29] do not account for communication denial environments, which is the primary concern of this article.

To address the aforementioned challenges, this article proposes a distributed DRL-based algorithm for the flocking control of a UAV swarm. The algorithm is formulated as a sequential decision problem for each UAV in an RL framework. It utilizes the soft actor-critic (SAC) method, a state-of-the-art off-policy actor-critic DRL algorithm. The main contributions of this work are as follows.

1) A distributed DRL-based algorithm is proposed to enable the UAV swarm to have robust generalization performance in unknown environments, improving the ability of the UAV swarm to perform missions in complex environments. The UAV swarm flocking control policy is trained in continuous state and action spaces without requiring precise models of the environment and UAVs. In addition, a CTDE framework is used, where the experience of all UAVs is utilized for training a distributed, shared policy network. This reduces the burden of online computation by performing the computationally resource-intensive training process offline.
2) The flocking control is considered in communication denial environments, where the UAVs can only sense other UAVs within their limited field of view through onboard sensors to achieve coordination. A centralized critic network augmented with extra information about the entire UAV swarm is proposed to facilitate training, which can improve the UAV swarm flocking performance when the UAVs have limited perception capability.
3) Collision avoidance between UAVs is implemented as an innate capability in the DRL framework, allowing the UAV swarm to balance collision avoidance and flock topology maintenance. By interacting with the environment, the UAVs can learn to cooperate and navigate complex environments as a flock without inter-UAV collisions.

II. Preliminaries

A. Problem Scenario

We examine a scenario where a swarm of UAVs, denoted as $\mathcal{U}(u_i \in \mathcal{U}, i = 1, 2, \ldots, N)$, consisting of N identical UAVs,

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

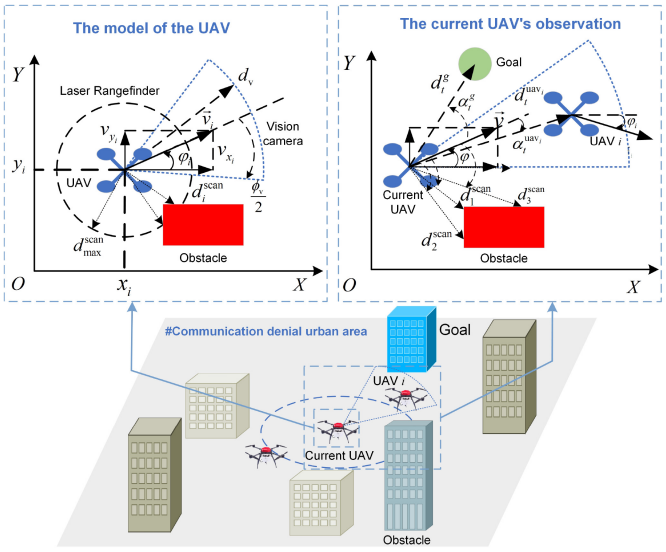

Fig. 1. Problem scenario considered in this article.

is tasked with scouting a crucial building in an urban environment. To minimize the risk of detection, the UAV swarm must maintain a tight formation and complete the mission as quickly as possible. Furthermore, the mission area is a communication-denied zone, in which inter-UAV communication could compromise the UAVs' movements. In this scenario, the UAVs can only detect and coordinate with other UAVs within their visual field, using onboard sensors, such as a laser rangefinder and a vision camera. The problem scenario is depicted in Fig. 1.

B. UAV Kinematics Model

The UAVs are assumed to fly at a fixed altitude. Thus, the kinematics model of the UAV can be modeled in a 2-D plane

$$\begin{cases} \dot{x}_i = v_{x_i} \\ \dot{y}_i = v_{y_i} \\ \dot{v}_{x_i} = \left(v_{x_i}^c - v_{x_i}\right)/\tau_v \\ \dot{v}_{y_i} = \left(v_{y_i}^c - v_{y_i}\right)/\tau_v \end{cases} \tag{1}$$

where (x_i, y_i), (v_{x_i}, v_{y_i}), and $(v_{x_i}^c, v_{y_i}^c)$, respectively, denote the position, velocity, and command velocity of the UAV i in a 2-D Cartesian coordinate system, and τ_v is the time constant related to the dynamics of the UAV.

C. UAV Perception System Model

The sensing range of the laser rangefinder is 0–360 degrees and can provide 36 distance measurements, denoted by $\mathbf{d}^{\text{scan}}$, with a maximum range of $d_{\max}^{\text{scan}}$. On the other hand, the vision camera has a perception angle of ϕ_{v} and a perception distance of d_{v}. It is important to note that the camera is also able to adjust its orientation to align with the velocity direction of the UAV, making it necessary to control only the position of the UAV and not its orientation. The direction of the velocity of UAV i is represented by the angle made with the x-axis, denoted by φ_i. The UAV model is illustrated in Fig. 1.

D. Problem Formulation

Formally, we address the problem of flocking control for a UAV swarm, denoted as $\mathcal{U}$, operating in an environment with M building obstacles, denoted as $\mathcal{B}(b_i \in \mathcal{B}, i = 1, 2, \ldots, M)$. At each time step t, each UAV i perceives the state of the environment s_t through its onboard sensors and generates an observation $o_i^t \sim O(s_t)$, where $O(s_t)$ is the observation function. The UAV then takes an action a_i^t according to its flocking control policy $\pi(a_i^t|o_i^t)$. The control objectives of policy $\pi(a_i^t|o_i^t)$ are two fold.

1) To enable each UAV in the swarm to navigate from the starting area to the goal area in the shortest possible time while avoiding collisions with obstacles and other UAVs.
2) To enable each UAV to maintain proximity to the center of the swarm, thereby forming a compact flock with the other UAVs.

This problem can be formulated as the following optimization problem:

$$\begin{aligned} \arg\min_{\pi} \ & \mathbb{E}\left[t^g|\pi\right] + \beta\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\bar{d}_i^c|\pi\right] \\ \text{s.t.} \ & \left\|\mathbf{p}_i - \mathbf{p}_{\text{g}}\right\| \le d_{\max}^g, \quad i = 1, 2, \ldots, N \\ & \left\|\mathbf{p}_i - \mathbf{p}_{i,j}^b\right\| \ge d_{\min}^b, \quad i = 1, 2, \ldots, N, j = 1, 2, \ldots, M \\ & \left\|\mathbf{p}_i - \mathbf{p}_j\right\| \ge d_{\min}^b, \quad i, j = 1, 2, \ldots, N, i \ne j \\ & \mathbf{p}_i^t = \mathbf{p}_i^{t-1} + \Delta t \mathbf{a}_i^{t-1}, \quad i = 1, 2, \ldots, N \end{aligned} \tag{2}$$

where t^g represents the time taken by the first UAV in the swarm to reach the goal area. The position of UAV i is represented by $\mathbf{p}_i = (x_i, y_i)$, and the center and radius of the circular goal area are represented by $\mathbf{p}_{\text{g}} = (x_g, y_g)$ and $d_{\max}^g$, respectively. The position of the closest point of obstacle j to UAV i is represented by $\mathbf{p}_{i,j}^b$, and $d_{\min}^b$ represents the minimum safe distance for collision avoidance. The average distance between UAV i and the center of the flock while the UAV swarm moves from the starting area to the goal area is represented by $\bar{d}_i^c$, and the positions of UAV i at times t and $t-1$ are represented by $\mathbf{p}_i^t$ and $\mathbf{p}_i^{t-1}$, respectively. The command of the velocity of UAV i at time $t-1$ is represented by $\mathbf{a}_i^{t-1}$, and the time step is represented by Δt. The weighting factor $\beta(\beta > 0)$ is used to balance the tradeoff between the optimality of time and the compactness of the flock.

The value $\bar{d}_i^c$ is calculated using the following equation:

$$\bar{d}_i^c = \frac{1}{t^g}\int_0^{t^g} d_i^c(t)dt \tag{3}$$

where $d_i^c(t)$ represents the distance between the UAV i and the center of the flock at time t, which is calculated as follows:

$$\begin{cases} d_i^c(t) = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \\ x_c = \frac{1}{N}\sum_{i=1}^{N} x_i \\ y_c = \frac{1}{N}\sum_{i=1}^{N} y_i. \end{cases} \tag{4}$$

As expressed in (2), the optimization objective contains both time-optimal and flock compactness metrics, and the constraints are the goal area constraints, collision avoidance

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

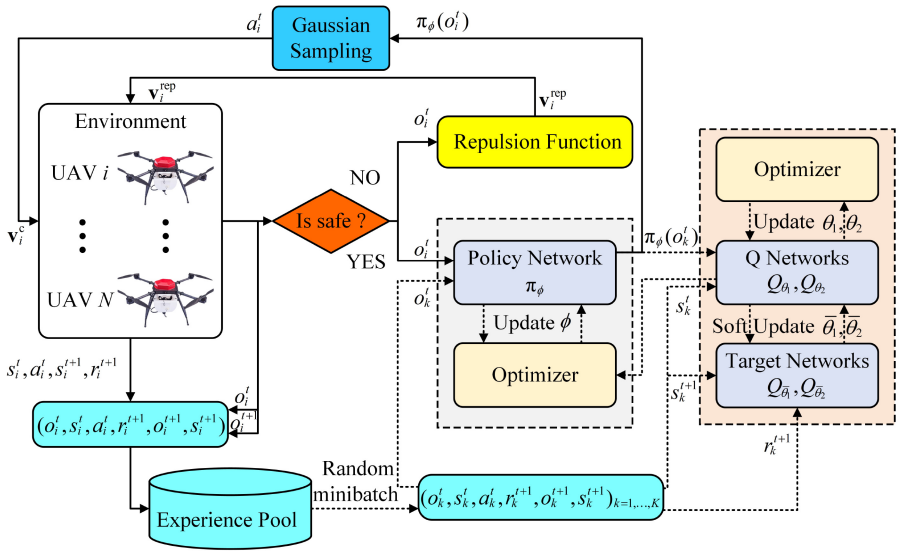

Fig. 2. System architecture.

constraints, and UAV kinematics constraints. In the next section, we translate the above optimization problem into a sequential decision problem in an RL framework and use a DRL method to solve it.

III. Approach

A. Overview

In this study, we investigate the use of a DNN, referred to as π_ϕ, as a control policy for a swarm of UAVs in flocking behavior. The policy is learned through interaction with the environment, as described in previous research by [20]. The system's architecture is depicted in Fig. 2. As shown, the flocking control policy π_ϕ is trained under a CTDE framework.

During the execution process, each UAV utilizes its perception system to obtain observations o_i^t of the environment and assess the potential risk of collisions. If a risk is detected, the UAV uses a repulsion function to calculate the necessary collision avoidance control command $\mathbf{v}_i^{\text{rep}}$. On the contrary, if the environment is considered safe, the policy network π_ϕ is used to calculate the flocking control command $\mathbf{v}_i^c$. The integration of the repulsion function ensures the safety of the UAVs while maintaining the integrity of the flock topology.

In the training process, the experiences of all UAVs are utilized to train a distributed, shared policy network. Furthermore, augmented states s_k^t and s_k^{t+1} that contain additional information about the entire swarm are utilized to train Q-value networks Q_{θ_1} and Q_{θ_2}, which help to train the policy network π_ϕ. The various components and details of the proposed approach are discussed in more detail in the following sections.

B. Reinforcement Learning

RL is a class of machine-learning methods used to solve sequential decision-making problems. Typically, an RL problem can be formulated as a Markov decision process (MDP), defined by a tuple $\langle S,\ A,\ P(s'|s,a),\ R(s',s,a),\gamma\rangle$, where S is the state space, A is the action space, $P(s'|s,a)$ is the state-transition model of the environment, $R(s',s,a)$ is the reward function, and $\gamma(0<\gamma<1)$ is a discount factor. At time t, an RL agent obtains state $s_t \in S$ from the environment and takes action $a_t \in A$ according to its policy $a_t \sim \pi(a_t|s_t)$. Subsequently, the environment state is changed to $s_{t+1} \in S$ based on the state-transition model $s_{t+1} \sim P(s'|s,a)$, and the agent receives a reward $r_{t+1} \in R(s',s,a)$. The RL agent learns the optimal policy $a_t \sim \pi^*(a_t|s_t)$ by interacting with the environment to maximize the long-term cumulative reward

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}. \tag{5}$$

In the RL framework, the value function of a state s under a policy π is defined as follows:

$$\begin{aligned} v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\ &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right] \end{aligned} \tag{6}$$

where the notation "$\doteq$" represents an equality relationship that is true by definition.

Similarly, the action-value function for a policy π is defined as follows:

$$\begin{aligned} Q_\pi(s,a) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] \\ &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right]. \end{aligned} \tag{7}$$

C. Soft Actor-Critic

We used the SAC [32], [33], a state-of-the-art off-policy actor-critic RL method, to solve the above problem. The SAC is based on the maximum entropy RL framework, where its objective is to learn a policy $\pi(a_t|s_t)$ that maximizes the following objective:

$$\pi^* = \arg\max_\pi \sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t) + \alpha\mathcal{H}(\pi(\cdot \mid s_t))] \tag{8}$$

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

where ρ_π denotes the trajectory distribution of the action of the state induced by a policy $\pi(a_t|s_t)$, $\mathcal{H}(\pi(\cdot \mid s_t))$ denotes the entropy of the policy $\pi(a_t|s_t)$, and α is the temperature parameter that balances the importance of the reward and the entropy term.

In the policy evaluation step of the SAC, the soft state value function is given by

$$V(s_t) = \mathbb{E}_{a_t\sim\pi}\big[Q(s_t, a_t) - \alpha \log \pi(a_t \mid s_t)\big] \tag{9}$$

where the soft Q-value $Q(s_t, a_t)$ can be computed by

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}\sim\rho}\big[V(s_{t+1})\big]. \tag{10}$$

In practical applications, the Q-function $Q_\theta(s_t, a_t)$ and a policy $\pi_\phi(a_t|s_t)$ can be approximated by neural networks with parameters θ and ϕ, respectively. The soft Q-function parameters can be trained to minimize the following loss:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}\Bigg[\frac{1}{2}\big(Q_\theta(s_t, a_t) - \big(r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}\sim\rho}\big[V_{\bar\theta}(s_{t+1})\big]\big)\big)^2\Bigg] \tag{11}$$

where $\mathcal{D}$ denotes the dataset, such as a reply buffer, and $\bar\theta$ represents the parameters of the target Q function, which can stabilize the training.

The policy parameters can be learned by minimizing the following loss function:

$$J_\pi(\phi) = \mathbb{E}_{s_t\sim\mathcal{D}}\big[\mathbb{E}_{a_t\sim\pi_\phi}\big[\alpha \log\big(\pi_\phi(a_t \mid s_t)\big) - Q_\theta(a_t, s_t)\big]\big]. \tag{12}$$

According to [32], the temperature parameter α can be automatically adjusted by minimizing the following loss:

$$J(\alpha) = \mathbb{E}_{a_t\sim\pi_t}\big[-\alpha \log \pi_t(a_t \mid s_t) - \alpha\bar{\mathcal{H}}\big] \tag{13}$$

where $\bar{\mathcal{H}}$ is the target entropy of the policy $\pi_t(a_t \mid s_t)$.

D. Ingredients of SAC

1) Observation Space: At the time t, the observation of each UAV $o_t = [o_t^g, o_t^{\text{scan}}, o_t^{\text{cam}}]$ consists of the relative position of the target o_t^g, partial measurements of the laser rangefinder o_t^{scan}, and the states of the other UAVs within the visual field o_t^{cam}. The current UAV observation is illustrated in Fig. 1.

The observation $o_t^g = [d_t^g, \alpha_t^g]$ represents the position of the goal in the UAV's heading coordinate system, where d_t^g represents the distance between the goal and the current UAV's position and α_t^g represents the angle between the vector pointing to the goal position from the UAV's current position and the current UAV's heading. The observation o_t^g is normalized as follows:

$$d_t^g = \begin{cases} d_t^g/d_{\text{env}}, & \text{if } d_t^g < d_{\text{S}} \\ 1.0, & \text{else} \end{cases} \tag{14}$$

$$\alpha_t^g = \alpha_t^g/\pi \tag{15}$$

where d_{S} is a constant related to the size of the environment.

The observation $o_t^{\text{scan}} = [d_1^{\text{scan}}, \alpha_1^{\text{scan}}, d_2^{\text{scan}}, \alpha_2^{\text{scan}}, d_3^{\text{scan}}, \alpha_3^{\text{scan}}]$ represents the three shortest laser rangefinder measurements and their angles relative to the current UAV's heading. Observations o_t^{scan} are arranged in ascending order of the distance $d_i^{\text{scan}}(i = 1, 2, 3)$, that is, d_1^{scan} is the shortest distance. The observation o_t^{scan} is normalized as follows:

$$d_i^{\text{scan}} = \begin{cases} d_i^{\text{scan}}/d_{\max}^{\text{scan}}, & \text{if } d_i^{\text{scan}} < d_{\max}^{\text{scan}} \\ 1.0, & \text{else.} \end{cases} \tag{16}$$

The normalization of α_i^{scan} is the same as in (15).

The observation $o_t^{\text{cam}} = [o_t^{\text{uav}_1}, o_t^{\text{uav}_2}, o_t^{\text{uav}_3}]$ represents the states of the three closest UAVs within its visual field, where $o_t^{\text{uav}_i} = [d_t^{\text{uav}_i}, \alpha_t^{\text{uav}_i}, \Delta\varphi^{\text{uav}_i}](i = 1, 2, 3)$ represents the state of UAV i concerning the current UAV. $d_t^{\text{uav}_i}$ represents the distance between UAV i and the current UAV, $\alpha_t^{\text{uav}_i}$ represents the angle between the vector pointing to the position of UAV i from the current UAV's position and the current UAV's heading, and $\Delta\varphi^{\text{uav}_i}$ represents the heading angle difference between UAV i and the current UAV, that is, $\Delta\varphi^{\text{uav}_i} = \varphi - \varphi_i$. If the number of UAVs observed is less than 3, the corresponding position is filled with $o_t^{\text{uav}_i} = [1, 0, 0]$. The observation o_t^{cam} is normalized as follows:

$$d_t^{\text{uav}_i} = \begin{cases} d_t^{\text{uav}_i}/d_{\text{v}}, & \text{if } d_t^{\text{uav}_i} < d_{\text{v}} \\ 1.0, & \text{else.} \end{cases} \tag{17}$$

The normalization of $\alpha_t^{\text{uav}_i}$ and $\Delta\varphi^{\text{uav}_i}$ are the same as (15).

2) Centralized Critic: In this study, we adopt the CTDE framework, as inspired by the work of [30], to train a decentralized policy and a centralized critic. The use of additional information is used to facilitate the training process. Specifically, the critic is augmented with the state of the center of the UAV swarm, $s_t^c = [d_t^c, \alpha_t^c, \Delta\varphi_t^c]$, where d_t^c represents the distance between the center of the UAV swarm and the current position of the UAV, α_t^c represents the angle between the vector that points to the center of the UAV swarm from the current position of the UAV and the direction of the current UAV, and $\Delta\varphi_t^c$ represents the difference between the direction of the current UAV and the average heading of the UAV swarm, which is calculated as follows:

$$\Delta\varphi_t^c = \varphi - \bar\varphi,\ \ \bar\varphi = \frac{1}{N}\sum_{i=1}^{N}\varphi_i. \tag{18}$$

Also, for the centralized critic, all UAVs have an unlimited visual field angle, that is, $\phi_{\text{v}} = 2\pi$ rad. Thus, the local observation $o_t^{\text{cam}} = [o_t^{\text{uav}_1}, o_t^{\text{uav}_2}, o_t^{\text{uav}_3}]$ is replaced with an improved state $s_t^{\text{cam}} = [s_t^{\text{uav}_1}, s_t^{\text{uav}_2}, s_t^{\text{uav}_3}]$, representing the states of the three closest UAVs within the visual field of the current UAV, where $s_t^{\text{uav}_i}$ and $o_t^{\text{uav}_i}$ have the same meaning.

Overall, the augmented state for the critic consists of o_t^g, o_t^{scan}, s_t^{cam}, and s_t^c, and is denoted as $s_t = [o_t^g, o_t^{\text{scan}}, s_t^{\text{cam}}, s_t^c]$. As in observation o_t, the augmented state s_t is normalized to be in the interval $[-1, 1]$.

3) Action Space: The action space is a set of limited command velocities in a continuous space. The command velocities of the UAV i are the translational velocities in the 2-D Cartesian coordinate system, that is, $\mathbf{v}_i^c = [v_{x_i}^c, v_{y_i}^c]$, $v_{x_i}^c, v_{y_i}^c \in [-v_{\max}, v_{\max}]$, where $v_{\max}$ is the maximum speed in one dimension. The output of the policy neural network is the change in the direction and speed of the UAV, indicated by

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

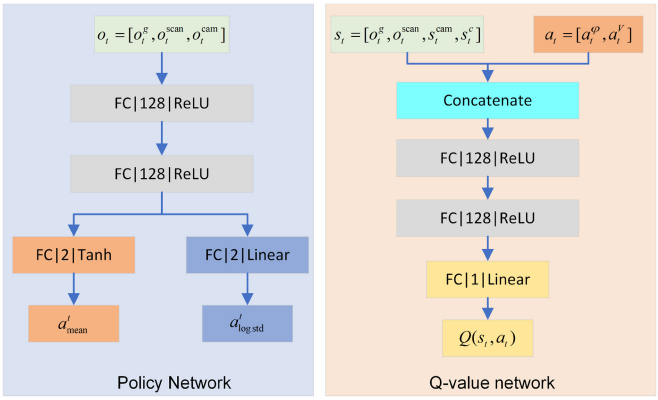

Fig. 3. DNN architectures.

$\Delta\varphi_i$ and V_i, respectively. Thus, the command velocities of the UAV i are calculated as follows:

$$\begin{cases} v^c_{x_i} = V_i \cos(\varphi_i + \Delta\varphi_i) \\ v^c_{y_i} = V_i \sin(\varphi_i + \Delta\varphi_i). \end{cases} \tag{19}$$

4) Network Architecture: We used two DNNs to approximate the Q-value and policy functions. The policy network maps the observation o_t to the action a_t, and the Q-value network maps the concatenation of the augmented state s_t and the action a_t to the Q-value $Q(s_t, a_t)$. Fig. 3 shows both the DNN architectures.

We used three fully connected neural network layers to approximate the policy network. Each layer has 128, 128, and 4 nodes, respectively. The first two layers have rectified linear units (ReLUs) as the activation function. The third output layer has two different outputs with different activations: 1) a hyperbolic tangent (tanh) is used to constrain the mean of the action a^t_{mean} in $[-1, 1]$ and 2) a linear function is used to output the log standard deviation a^t_{logstd}. Subsequently, the action $a_t = [a_t^\varphi, a_t^V]$ is sampled from a Gaussian distribution $\mathcal{N}(a^t_{\text{mean}}, \exp(a^t_{\text{logstd}}))$. Finally, the change in the heading and the speed of the UAV are calculated as follows:

$$\begin{cases} \Delta\varphi = a_t^\varphi * \pi/2, \ \ \Delta\varphi \in [-\pi/2, \pi/2] \\ V = a_t^V + 2, \ \ V \in [1, 3]. \end{cases} \tag{20}$$

The Q-value network is similar to the policy network except that its third output layer has one node with a linear activation function.

5) Reward Function: The design of the reward function is used to encourage the UAV swarm to learn a flocking control policy that meets (2). Thus, a reward function is designed to achieve this objective

$$r_t = r_t^g + r_t^c + r_t^f + r_t^v \tag{21}$$

where r_t is the reward received by the UAV at the time step t, and consists of four terms, namely, the reward for reaching the goal r_t^g, the obstacle avoidance reward r_t^c, the flocking control reward r_t^f, and the penalty reward r_t^v, related to changes in velocity commands.

The goal-reaching reward r_t^g awards the UAV for approaching and reaching its goal, and is designed as follows:

$$r_t^g = \begin{cases} 20.0, & \text{if } d_t^g < d_{\max}^g \\ 0.2\big(d_{t-1}^g - d_t^g\big) + 0.02\big(\pi/2 - |\alpha_t^g|\big) - 0.02, & \text{else.} \end{cases} \tag{22}$$

The obstacle avoidance reward r_t^c penalizes the UAV for collisions with the obstacles and is designed as follows:

$$r_t^c = \begin{cases} -0.2, & \text{if } \ 2 < d_1^{\text{scan}} \le 3 \\ -0.5, & \text{elif } \ 0.5 < d_1^{\text{scan}} \le 2 \\ -1.0, & \text{elif } \ d_1^{\text{scan}} \le 0.5 \\ 0, & \text{else.} \end{cases} \tag{23}$$

The flocking control reward r_t^f encourages the UAV swarm to maintain a compact and consistent flock and is designed as follows:

$$r_t^f = 0.05\big(1 - \Delta\varphi_t^c/\pi\big) + 0.05\big(1 - d^c(t)/3\big). \tag{24}$$

The penalty reward r_t^v encourages the UAV to move smoothly and is designed as follows:

$$r_t^v = -0.02|a_{t-1} - a_t|. \tag{25}$$

In the design of the reward function, we do not consider the case where the UAV collides with other UAVs. We consider collision avoidance between UAVs as an instinct, and we design a repulsion function to implement it in Section III-E.

E. Repulsion Function

We designed a repulsion function to implement the collision avoidance capability between UAVs and obstacles based on the work of [7]. Although the learned policy can avoid collisions with obstacles, in general, adding a repulsion term for obstacles can provide more assurance for the UAVs in terms of safety.

First, a linear distance-dependent central velocity term is proposed

$$\mathbf{v}_{ij}^r = \begin{cases} p^{\text{rep}}\big(r_0^{\text{rep}} - \|\mathbf{p}_{ij}\|\big)\frac{\mathbf{p}_i - \mathbf{p}_j}{\|\mathbf{p}_{ij}\|}, & \text{if } \|\mathbf{p}_{ij}\| < r_0^{\text{rep}} \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

where p^{rep} is the linear gain of repulsion, r_0^{rep} is the maximum interaction range, and $\|\mathbf{p}_{ij}\|$ is the distance between the UAVs i and j.

This central velocity term will cause oscillations because it is only related to the relative positions of the UAVs. To minimize the oscillations, a tangential velocity term $\mathbf{v}_{ij}^t$ is proposed that satisfies the following constraints:

$$\left\|\mathbf{v}_{ij}^t\right\| = \left\|\mathbf{v}_{ij}^r\right\|, \quad \mathbf{v}_{ij}^t \mathbf{v}_{ij}^r = 0, \quad \big(\mathbf{p}_{\text{g}} - \mathbf{p}_i\big)\mathbf{v}_{ij}^t > 0. \tag{27}$$

Equation (27) means that the magnitude of the tangential velocity term $\mathbf{v}_{ij}^t$ is equal to that of the central velocity term $\mathbf{v}_{ij}^r$ and its direction is perpendicular to $\mathbf{v}_{ij}^r$, with the angle between $\mathbf{v}_{ij}^t$ and $(\mathbf{p}_{\text{g}} - \mathbf{p}_i)$ being an acute angle. Thus, the final repulsion velocity between UAVs i and j can be formulated as follows:

$$\mathbf{v}_{ij}^{\text{rep}} = \mathbf{v}_{ij}^r + \mathbf{v}_{ij}^t. \tag{28}$$

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

TABLE I
PARAMETERS OF ALGORITHM 1

Parameters	Values	Parameters	Values	Parameters	Values
$\mathcal{C}$	1e5	K	128	λ_α	1e-3
$T_{\max}$	500	γ	0.99	α	0.2
N	5	λ_Q	1e-3	$\bar{\mathcal{H}}$	-2
d_{safe}	2.0m	λ_π	5e-5	τ	0.01

TABLE II
PARAMETERS OF THE UAV MODEL

Parameters	Values	Parameters	Values	Parameters	Values
τ_v	1.0s	d_{v}	5.0m	$v_{\max}$	3m/s
$d^{\text{scan}}_{\max}$	10.0m	$d^{g}_{\max}$	1.0m	p^{rep}	1.0
ϕ_{v}	2π, 0.5π	$d^{b}_{\min}$	0.5m	r^{rep}_0	5.0

For obstacles, the point closest to the UAV i on the obstacle surface is considered a virtual UAV $\mathcal{V}$. Finally, the total repulsion term for UAV i concerning the other UAVs and obstacles is calculated as follows:

$$\mathbf{v}_i^{\text{rep}} = \sum \mathbf{v}_{ij}^{\text{rep}}, j = 1, 2, \ldots, N, \mathcal{V}, i \neq j. \tag{29}$$

F. Flocking Control Algorithm

In this section, we present a flocking control algorithm for a UAV swarm that combines the SAC algorithm and the repulsion function. The SAC algorithm is utilized to learn a shared and distributed policy that can control each UAV's flock as a compact group while avoiding obstacles. The repulsion function provides an instinctive collision avoidance capability for UAVs. The policy is trained using the CTDE paradigm. During the training stage, each UAV independently observes the state of the environment, performs actions, and then uses the experiences of all UAVs to train the networks. The workflow of the flocking control algorithm is outlined in Algorithm 1.

As summarized in Algorithm 1, we utilize two soft Q-value networks to mitigate the positive bias in the policy update step, as previously proposed in [32] and [34]. These networks are trained independently and the minimum of their values is used to update the policy network in (12) using the Adam optimizer, as outlined in [35]. The training process alternates between collecting the experiences of all UAVs from the environment with the current strategy and updating the networks and the temperature parameter using a minibatch of experiences randomly sampled from a replay buffer. The execution strategy depends on the minimum values of d_t^{uav1} and d_1^{scan}. If the minimum value is greater than the safe distance d_{safe}, the command velocities are calculated using the policy network; otherwise, the command velocities are calculated using the repulsion function. This approach allows the flocking control policy to adapt to the UAVs' instinctive repulsion capability while improving the collected experience's quality.

Algorithm 1: SAC for UAV Flocking Control With Repulsion

Initialize policy network π_ϕ and Q-value network Q_{θ_1}, Q_{θ_2}
Initialize target Q-value network $Q_{\bar{\theta}_1}$, $Q_{\bar{\theta}_2}$, $\bar{\theta}_1 \leftarrow \theta_1$, $\bar{\theta}_2 \leftarrow \theta_2$
Initialize an empty replay buffer $\mathcal{D} \leftarrow \emptyset$ with size $\mathcal{C}$
for *episode = 1,2, ...,* **do**
  for t = *1,2, ...,* $T_{\max}$ **do**
    for *UAV* i = *1, 2, ...,* N **do**
      Observe the environment states o_i^t and s_i^t, select action $a_i^t \sim \pi_\phi(a_i^t|o_i^t)$
      if $\min(d_t^{\text{uav1}}, d_1^{\text{scan}}) > d_{\text{safe}}$ **then**
        Using equation (19) calculates the command velocities
      else
        Using equation (29) calculates the command velocities
      end
      Execute the command velocities, and then receive reward r_i^{t+1} and observe new environment states o_i^{t+1} and s_i^{t+1}
      Store transition $(o_i^t, s_i^t, a_i^t, r_i^{t+1}, o_i^{t+1}, s_i^{t+1})$ in $\mathcal{D}$
    end
    Sample a random minibatch of K transitions $(o_i^t, s_i^t, a_i^t, r_i^{t+1}, o_i^{t+1}, s_i^{t+1})$ from $\mathcal{D}$
    Update Q-value networks by minimizing the loss in (11),
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$
    Update the policy network by minimizing the loss in (12), $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
    Update the temperature parameter by minimizing the loss in (13),
    $\alpha \leftarrow \alpha - \lambda_\alpha \hat{\nabla}_\alpha J(\alpha)$
    Update the target network
    $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$ for $i \in \{1, 2\}$
  end
end

IV. Simulation Experiments and Results

A. Simulation Setup and Training Results

We evaluated the proposed flocking control algorithm through numerical simulations. The Q-value and policy networks were trained in a 2-D environment comprising five UAVs and three obstacles. The training environment was a rectangular area with a size of 80 m × 80 m. The networks were implemented using the PyTorch deep learning framework. Table I provides a summary of the parameters used in Algorithm 1, and Table II lists the parameters of the UAV model.

1) *Policy1:* The angle of the visual field of each UAV ϕ_{v} is set to 2π radians and there is no instinctive repulsion ability.
2) *Policy1-G:* The angle of view of the field of each UAV ϕ_{v} is set to 2π radians, and there is no instinctive repulsion ability.
3) *Policy2:* The angle of view of the field of each UAV ϕ_{v} is set to 0.5π radians, and there is no instinctive repulsion ability.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

4) *Policy2-G:* The angle of view of the field of each UAV ϕ_{v} is set to 0.5π radians, and there is no instinctive repulsion ability.
5) *Policy3:* The angle of the visual field of each UAV ϕ_{v} is set to 0.5π radians, and there is an instinctive repulsion ability.
6) *Policy3-G:* The visual field angle of each UAV ϕ_{v} is set to 0.5π radians, and there is an instinctive repulsion ability.
7) *Policy4-G:* The visual field angle of each UAV ϕ_{v} is set to 0.5π radians, and there is no instinctive repulsion ability, instead, the collision avoidance capability between UAVs is obtained using a learning method, that is, replacing d_1^{scan} in (23) with $\min(d_t^{\mathrm{uav1}}, d_1^{\mathrm{scan}})$.
8) *Policy5-G:* The visual field angle of each UAV ϕ_{v} is set to 0.5π radians, and there is instinctive repulsion ability, but it does not consider the term tangential repulsion, that is, (28) is changed to $\mathbf{v}_{ij}^{\mathrm{rep}} = \mathbf{v}_{ij}^{r}$.

For Policy1, Policy2, and Policy3, the critic is not augmented with additional information, that is, the observation of the critic network is the same as the observation of the policy network. For Policy1-G, Policy2-G, Policy3-G, Policy4-G, and Policy5-G, the critic is augmented with additional information as described in Section III-D2.

In training, the value of d_{S} is set to 80 m, and the time step Δt is set to 0.2 s. We used 5000 episodes to train the eight policies. At the beginning of each training episode, for a high-quality experience, the positions of the UAVs and the goal were randomly reset, and the positions of the obstacles were fixed, with the positions of the UAVs' center and the goal set as follows:

$$\begin{cases} x_c = d_{\mathrm{env}} \cos(\chi) + 40 \\ y_c = d_{\mathrm{env}} \sin(\chi) + 40 \\ x_g = d_{\mathrm{env}} \cos(\chi + \pi) + 40 \\ y_g = d_{\mathrm{env}} \sin(\chi + \pi) + 40 \end{cases} \tag{30}$$

where (x_c, y_c) represents the center of the UAVs, d_{env} is the distance value determining the distance between the UAVs and the goal area and is set to 40 m in training, χ is an angle sampled from $[-\pi, \pi]$ uniformly. The position of each UAV was randomly sampled from a circular area with a center (x_c, y_c) and a radius of 3 m.

The obstacles consist of two rectangles and one circle, and their centers are located at (10, 40) m, (70, 40) m, and (40, 40) m, respectively. Fig. 4 shows the training results under the eight policies.

As illustrated in Fig. 4, all eight policies, except Policy4-G, can obtain stable rewards after 1000 episodes (Policy4-G can obtain stable rewards after 3000 episodes) and maintain this performance up to 5000 episodes. This means that all eight trained flocking control policies can allow the UAV swarm to navigate to the target area in the training environment.

In particular, Policy1-G can obtain more rewards than Policy1 in the early stages of the training, indicating that using a centralized critic can enable the policy to achieve better flocking control rewards. As training progresses, Policy1 and Policy1-G receive similar rewards, suggesting that Policy1 can effectively control the UAV swarm to form a tight flock and simultaneously infer the states of the flock center from the states of neighboring UAVs. Thus, the effect of the centralized critic on the training of Policy1-G becomes insignificant. Policy2-G and Policy3-G can receive more rewards during training than Policy2 and Policy3, respectively. This is because Policy2, Policy3, Policy2-G, and Policy3-G have limited visual fields, and the use of additional information from the centralized critic can aid these policies in determining global states. As a result, Policy2-G and Policy3-G can obtain better flocking control rewards than Policy2 and Policy3. From this comparison, we can infer that when the UAV has limited perception, the use of a centralized critic has a more significant impact on the training of the policy, leading to better rewards for the UAV.

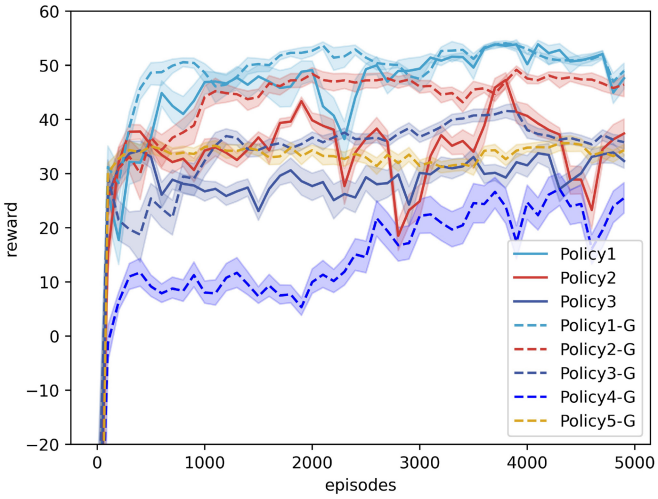

Fig. 4. Curves of the rewards for different policies in training. (Note: The rewards are calculated every 100 episodes and the size of the confidence interval is set to 0.95.)

Furthermore, after the policies have learned the flocking control ability in the final stages of the training, for policies augmented with additional information, we find that Policy1-G has the most rewards, whereas Policy3-G has the least. This is because all three policies can control the swarm of UAVs to reach the target area and avoid collisions with obstacles. However, Policy1-G has the widest visual field, which enables the UAVs to obtain more rewards for flocking control. In addition, Policy3-G controls each UAV to avoid collisions with other UAVs and ensures that each UAV is far away from the center of the UAV swarm, thus obtaining fewer rewards. Policy2-G obtains fewer rewards than Policy1-G due to its limited visual field. Policy1, Policy2, and Policy3 have similar results.

In addition, compared to Policy5-G, Policy3-G can obtain more rewards during the training process. This indicates that the tangential repulsion term in instinctive repulsion can improve the flocking control performance of Policy3-G. However, compared to Policy3-G and Policy5-G, Policy4-G has a slower convergence rate throughout the training process and gets the least rewards after the training is stable. This suggests that learning collision avoidance between UAVs and compact flocking control ability is challenging through learning alone. However, this problem can be effectively

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

TABLE III
COMPARISON RESULTS OF DIFFERENT POLICIES IN THE TRAINING ENVIRONMENT

	$d_{\min}^{\text{obs}}$(m)	$d_{\min}^{\text{uav}}$(m)	d_{c}(m)	V_{error}^{c}(m/s)	φ_{error}^{c}(rad)	V(m/s)	t^g(s)	Success rate
Policy1	3.265±0.772	0.007±0.006	1.112±0.891	0.232±0.171	0.146±0.151	2.005±0.066	41.895±4.723	88.8%
Policy1-G	3.709±0.676	0.013±0.010	**0.804±0.118**	0.177±0.032	0.121±0.105	1.984±0.079	44.118±2.534	93.6%
Policy2	3.142±0.645	0.190±0.165	2.945±0.736	0.203±0.053	0.156±0.136	1.771±0.048	44.079±1.117	87.2%
Policy2-G	3.212±0.689	0.046±0.057	1.275±0.394	**0.105±0.051**	**0.081±0.113**	2.169±0.065	38.048±1.945	**97.0%**
Policy3	**4.272±1.241**	1.581±0.236	3.985±0.651	0.315±0.056	0.297±0.270	1.913±0.113	40.611±3.133	87.6%
Policy3-G	3.549±0.937	1.627±0.276	3.393±0.610	0.268±0.067	0.159±0.219	2.052±0.076	36.709±1.434	93.8%
Policy4-G	3.448±0.660	1.047±0.380	3.001±0.295	0.121±0.033	0.086±0.125	2.206±0.041	34.856±1.202	65.4%
Policy5-G	2.598±0.391	1.742±0.162	3.476±0.736	0.216±0.060	0.162±0.155	2.228±0.070	33.326±1.573	88.6%
PolicyD-G	3.365±1.194	1.584±0.211	3.209±0.436	0.288±0.053	0.185±0.192	2.075±0.078	38.108±2.124	88.2%
PolicyP-G	2.529±1.316	1.572±0.367	3.438±1.104	0.277±0.070	0.176±0.179	1.959±0.077	40.841±2.000	82.6%
Policy-MA	1.768±0.052	**1.780±0.524**	5.055±0.529	0.477±0.085	0.367±0.069	**2.595±0.036**	34.929±0.757	82.2%
Policy-OP	2.204±0.599	1.141±0.532	5.292±1.485	0.678±0.065	0.456±0.223	2.486±0.056	**34.771±1.110**	85.6%

solved by considering collision avoidance among UAVs as an instinctive capability, as shown in Policy3-G.

In the next section, we compare the performance of the proposed policies with various test cases.

B. Comparison Tests

This section compares our policies using learning-based, traditional, and nonlearning methods.

1) The policy is trained using DDPG [36], which is an actor-critic algorithm based on the deterministic policy gradient. In addition, the critic is augmented with additional information in training and UAVs have instinctive repulsion ability. We name this policy PolicyD-G.
2) The policy is trained by proximal policy optimization (PPO) [37], which is a policy optimization method with the stability and reliability of the trust region. In this study, the PPO is implemented in an actor-critic style with the critic augmented with additional information in training. In addition, UAVs have instinctive repulsion, which is why we called this policy PolicyP-G.
3) The policy is trained according to [11], where the flocking behavior is generated by constructing collective potentials. We name this policy Policy-MA.
4) The policy is trained according to [7], where the flocking problem is solved in an optimization way. We call this policy Policy-OP.

We use the following two metrics to evaluate the collision avoidance capability of UAVs.

1) The minimum distance $d_{\min}^{\text{obs}}$ between the UAVs and the obstacles in an episode, calculated by

$$\begin{aligned} d_{\min}^{\text{obs}} &= \min\left(\mathbf{D}^{\text{obs}}\right) \\ \mathbf{D}^{\text{obs}} &= \left\{d | d = d_t^{\text{obs}}(u_i),\ i = 1, 2, \ldots, N,\ t = 1, \ldots, T_{\max}\right\} \end{aligned} \tag{31}$$

where $d_t^{\text{obs}}(u_i)$ represents the minimum distance between UAV i and obstacles at time step t.

2) The minimum distance $d_{\min}^{\text{uav}}$ between the UAVs in an episode, calculated by

$$\begin{aligned} d_{\min}^{\text{uav}} &= \min\left(\mathbf{D}^{\text{uav}}\right) \\ \mathbf{D}^{\text{uav}} &= \left\{d | d = d_t^{\text{uav}}(u_i),\ i = 1, 2, \ldots, N,\ t = 1, \ldots, T_{\max}\right\} \end{aligned} \tag{32}$$

where $d_t^{\text{uav}}(u_i)$ represents the minimum distance between the UAV i and the other UAVs at the time step t.

We use the following three metrics to evaluate the compactness and consistency of the UAV flock.

1) The average distance d_{c} from the UAVs to the center of the UAV swarm in an episode, which is calculated by

$$d_c = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T} d_i^c(t). \tag{33}$$

2) The average speed difference V_{error}^{c} between the UAVs and the center of the UAV swarm, calculated by

$$\begin{cases} V_{\text{error}}^{c} = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T}|V_i - V_c| \\ V_i = \sqrt{v_{x_i}^2 + v_{y_i}^2} \\ V_c = \sqrt{\left(\sum_{i=1}^{N} v_{x_i}\right)^2 + \left(\sum_{i=1}^{N} v_{y_i}\right)^2}. \end{cases} \tag{34}$$

3) The average heading deviation $\varphi_{\text{error}}^{c}$ between the UAVs and the center of the UAV swarm, calculated by

$$\varphi_{\text{error}}^{c} = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T}|\varphi_i - \bar{\varphi}|. \tag{35}$$

We use the following two metrics to evaluate the flying speed and optimal time of the UAV flock.

4) The average speed V of UAVs in an episode, calculated by

$$V = \frac{1}{T}\sum_{t=1}^{T} V_c. \tag{36}$$

5) The time t^g it took the first UAV to reach the target area.

Furthermore, we measure the success rate as the ratio of the number of tests that successfully reach the goal area without collision within the total time step $T_{\max}$ in each test scenario. We randomly tested each policy 500 times in the training environment using the same random seed. The average and standard deviation of the aforementioned metrics are listed in Table III.

As shown in Table III, Policy1-G, Policy2-G, and Policy3-G perform better than Policy1, Policy2, and Policy3, respectively, in almost all metrics. This indicates that the centralized critic can train a better policy for controlling the UAVs as a compact flock and navigating them from the starting area to the goal area without collision with obstacles. Policy1-G has the

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

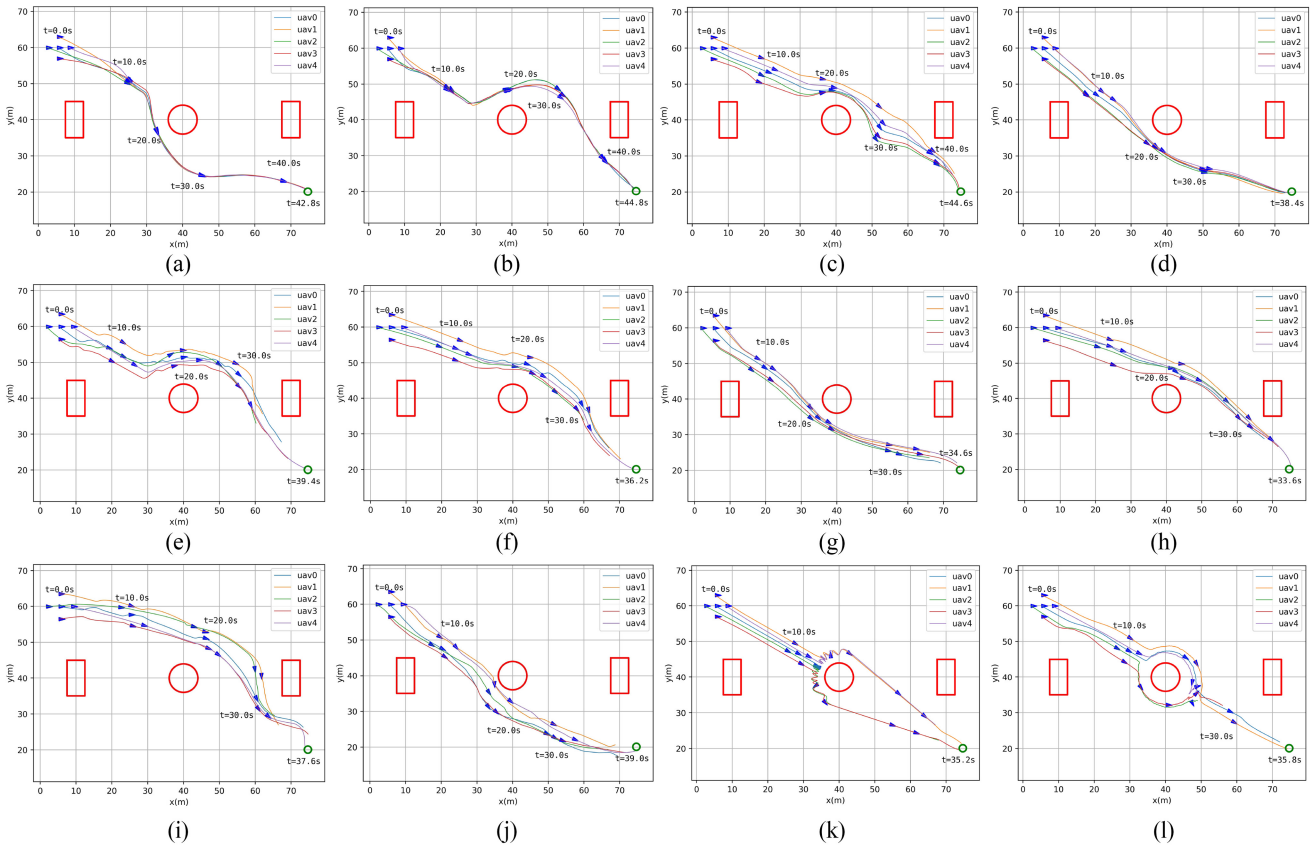

Fig. 5. Test case results for different policies in the training environment. (a) Policy1. (b) Policy1-G. (c) Policy2. (d) Policy2-G, (e) Policy3. (f) Policy3-G. (g) Policy4-G. (h) Policy5-G. (i) PolicyD-G. (j) PolicyP-G. (k) Policy-MA. (l) Policy-OP

lowest value of d_c as it has the maximum visual field and can ensure that UAVs approach the center of the UAV swarm more easily without considering collision avoidance. Policy2-G has the highest success rate of 97.0.

For all trained policies, except PolicyP-G and Policy5-G, the average minimum distance $d_{\text{min}}^{\text{obs}}$ is greater than 3 m, which is the safe distance designed in the reward function in (23). Compared to Policy3-G and PolicyD-G, PolicyP-G has a lower success rate with smaller values of $d_{\text{min}}^{\text{obs}}$ and $d_{\text{min}}^{\text{uav}}$, indicating that the UAVs controlled by Policy3-G are more likely to collide with obstacles as well as other UAVs. Furthermore, although PolicyD-G has a lower value of d_c, Policy3-G has lower values of V_{error}^c and φ_{error}^c and a higher success rate; therefore, Policy3-G has superior flock control performance.

Furthermore, compared to Policy3-G and Policy5-G, Policy4-G has the lowest values of V_{error}^c and φ_{error}^c, indicating that when the collision avoidance ability and the flocking control ability are simultaneously learned through the learning method, the learned policy can make the UAV swarm more compact and consistent without instinctive repulsion. However, Policy4-G also has the lowest success rate (65.4%) due to its inability to balance collision avoidance between UAVs and compact flocking control, resulting in collisions between UAVs. This can be observed in the fact that it has the smallest $d_{\text{min}}^{\text{uav}}$. Furthermore, the results of Policy5-G show that when there is no tangential repulsion term, both $d_{\text{min}}^{\text{obs}}$ and the success rate decrease, indicating that the tangential repulsion term can improve the UAVs' collision avoidance ability.

Furthermore, all trained policies have a lower average speed of the UAVs compared to Policy-MA and Policy-OP. One of the main reasons for this is that reducing the speed makes it easier to form a compact flock and obtain more flocking control rewards. Compared to Policy-MA and Policy-OP, Policy3-G has a higher success rate and lower values of d_c, V_{error}^c, and φ_{error}^c, indicating that it can enable UAVs to form a compact flock with high-velocity consistency. Policy-MA and Policy-OP have low success rates and high values of d_c due to their inability to achieve a balance between collision avoidance and flock maintenance, resulting in collisions with obstacles or sparse flocks; this is further proved by the low values of $d_{\text{min}}^{\text{obs}}$ under Policy-MA and Policy-OP. Fig. 5 illustrates the test case results for different policies in the training environment.

Fig. 5(a) and (b) show the trajectories of the UAVs controlled by Policy1 and Policy1-G, respectively, in an environment with three obstacles represented as two red rectangles and one red circle. The blue triangles indicate the positions of the UAVs at different times. The green circle indicates the target area. Policy1 and Policy1-G can enable the UAV swarm to form a compact flock and navigate it from the starting area to the goal area without collision with the obstacles.

Fig. 5(c) and (d) show the trajectories of the UAVs controlled by Policy2 and Policy2-G, respectively. Compared with Fig. 5(a) and (b), the UAV swarm is sparser because the UAVs have limited visual fields. Furthermore, compared to Fig. 5(c) and (d) show that the UAV swarm is more compact

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
COMPARISON RESULTS OF FIVE POLICIES UNDER FIVE DIFFERENT SCENARIOS

		$d_{\min}^{\text{obs}}$(m)	$d_{\min}^{\text{uav}}$(m)	d_c(m)	V_{error}^c(m/s)	φ_{error}^c(rad)	V(m/s)	t^g(s)	Success rate
S1	Policy3-G	10.000±0.000	1.876±0.099	3.117±0.277	0.104±0.019	0.110±0.178	2.227±0.018	33.653±0.682	100%
	PolicyD-G	10.000±0.000	1.804±0.118	3.191±0.275	0.140±0.013	0.172±0.160	2.137±0.014	36.849±0.236	100%
	PolicyP-G	10.000±0.000	1.678±0.211	3.172±0.388	0.167±0.014	0.179±0.211	2.015±0.021	39.848±0.999	100%
	Policy-MA	10.000±0.000	2.248±0.250	**2.200±0.188**	0.036±0.004	**0.026±0.101**	2.698±0.008	28.588±0.133	100%
	Policy-OP	10.000±0.000	**2.690±0.283**	2.517±0.014	**0.002±0.002**	0.027±0.073	**2.906±0.001**	**26.646±0.275**	100%
S2	Policy3-G	3.800±0.981	1.327±0.333	**4.135±0.339**	**0.359±0.055**	**0.174±0.199**	1.963±0.052	35.794±1.144	**85.6%**
	PolicyD-G	**4.212±1.101**	1.443±0.261	4.188±0.323	0.373±0.040	0.231±0.210	2.046±0.056	37.521±1.174	80.2%
	PolicyP-G	1.698±1.113	1.130±0.390	4.166±1.196	0.387±0.072	0.190±0.126	1.911±0.078	40.106±1.674	69.2%
	Policy-MA	1.723±0.089	1.230±0.464	5.090±0.723	0.495±0.070	0.396±0.097	2.614±0.044	33.263±1.040	82.0%
	Policy-OP	1.792±0.464	**1.583±0.536**	5.733±1.194	0.709±0.048	0.478±0.260	**2.628±0.037**	**32.729±0.782**	83.8%
S3	Policy3-G	2.234±0.667	1.609±0.215	4.694±0.796	**0.234±0.048**	**0.288±0.303**	2.317±0.060	**49.675±1.586**	**91.2%**
	PolicyD-G	1.640±0.589	1.222±0.329	4.730±1.037	0.309±0.075	0.350±0.253	2.434±0.113	51.355±3.646	84.0%
	PolicyP-G	1.168±0.976	1.124±0.437	5.331±1.811	0.381±0.080	0.330±0.311	1.890±0.087	63.142±2.213	76.6%
	Policy-MA	1.817±0.111	1.012±0.844	**4.242±1.378**	0.343±0.067	0.293±0.089	**2.584±0.064**	50.808±1.940	70.4%
	Policy-OP	**2.686±0.423**	**1.716±0.635**	7.043±2.461	0.594±0.122	0.524±0.314	2.484±0.126	53.229±3.997	75.8%
S4	Policy3-G	3.282±0.802	1.420±0.246	3.555±0.454	**0.279±0.058**	**0.171±0.191**	2.061±0.066	36.158±1.073	**91.6%**
	PolicyD-G	**3.448±1.157**	1.468±0.198	**3.543±0.388**	0.317±0.048	0.191±0.216	2.055±0.067	38.177±2.084	83.4%
	PolicyP-G	2.446±1.699	1.196±0.308	3.544±0.923	0.287±0.049	0.193±0.150	1.974±0.066	40.891±2.404	75.2%
	Policy-MA	1.700±0.088	**1.550±0.559**	5.118±0.480	0.482±0.079	0.371±0.066	**2.609±0.032**	**34.410±0.854**	74.4%
	Policy-OP	2.205±0.638	1.282±0.518	5.312±0.784	0.691±0.073	0.484±0.231	2.473±0.065	34.953±1.010	81.8%
S5	Policy3-G	**2.505±0.926**	1.547±0.259	**4.402±0.838**	**0.395±0.082**	**0.233±0.217**	1.907±0.104	37.701±2.917	**82.2%**
	PolicyD-G	2.122±0.941	1.314±0.349	4.737±1.221	0.403±0.081	0.311±0.193	1.922±0.100	40.557±3.044	79.2%
	PolicyP-G	2.049±1.190	1.267±0.203	4.691±0.886	0.408±0.048	0.323±0.156	1.952±0.064	41.633±2.539	72.8%
	Policy-MA	1.766±0.064	**1.612±0.689**	4.901±0.828	0.486±0.104	0.370±0.081	**2.585±0.044**	**34.975±1.189**	67.6%
	Policy-OP	2.077±0.688	1.296±0.575	5.160±0.501	0.655±0.068	0.512±0.238	2.313±0.160	38.519±3.934	78.2%

and takes less time to reach the goal area, which implies that the central critic has a positive effect on the policy when UAVs have limited perception capability.

Fig. 5(e) and (f) show the trajectories of the UAVs controlled by Policy3 and Policy3-G, respectively. Due to the repulsion function of UAVs, the flock becomes more sparse than in the case of Policy1, Policy1-G, Policy2, and Policy2-G. Similarly, the UAV swarm controlled by Policy3-G forms a more compact and consistent flock than that controlled by Policy3.

Fig. 5(g) and (h) show the trajectories of the UAVs controlled by Policy4-G and Policy5-G, respectively. Compared with Fig. 5(f), the UAV swarm controlled by Policy4-G has a more compact flock, and the UAV swarm controlled by Policy5-G is closer to the obstacles, which is consistent with the results in Table III.

Fig. 5(i) and (j) show the trajectories of the UAVs controlled by PolicyD-G and PolicyP-G, respectively. Compared with Fig. 5(f), the UAV swarm controlled by PolicyD-G has a more compact flock and takes more time to its goal area, and the UAV swarm controlled by PolicyP-G is closer to the obstacles.

Fig. 5(k) and (l) show the trajectories of the UAVs controlled by Policy-MA and Policy-OP, respectively. As shown in Fig. 5(k), the UAVs oscillate near the circular obstacle. In addition, the UAV swarm controlled by Policy-OP is divided into two parts when encountering an obstacle, as shown in Fig. 5(l), and the UAV swarm does not avoid obstacles as an entire. The lack of a practical obstacle avoidance strategy is the main reason for the poor flocking results of Policy-MA and Policy-OP when encountering obstacles.

C. Generalization Tests

We tested the generalization performance of Policy3-G in five different scenarios, all of which were different from the training scenario.

1) The first scenario (Scenario 1, S1) has five UAVs with no obstacles.
2) The second scenario (Scenario 2, S2) has nine UAVs and the environment is the same as the training environment; that is, there are three obstacles in the environment.
3) The third scenario (Scenario 3, S3) has five UAVs, and the environment is more complex, that is, it has nine obstacles. In Scenario 3, we set the value of d_{env} as 60 m.
4) The fourth scenario (Scenario 4, S4) considers the observation errors of the UAVs. In practice, it is difficult for the UAVs to accurately obtain the states of the other UAVs and the obstacles through vision cameras and laser rangefinders. For this reason, we added random errors to the observations of UAVs to simulate the real environment. Specifically, we add the Gaussian noise to the UAV observations o_t^{scan} and o_t^{cam}, that is, $d_i^{\text{scan}} = d_i^{\text{scan}} + 0.1\mathcal{N}(0,1)$, $d_t^{\text{uav}_i} = d_t^{\text{uav}_i} + 0.2\mathcal{N}(0,1)$, $\alpha_t^{\text{uav}_i} = \alpha_t^{\text{uav}_i} + 0.05\mathcal{N}(0,1)$, and $\Delta\varphi^{\text{uav}_i} = \Delta\varphi^{\text{uav}_i} + 0.05\mathcal{N}(0,1)$.
5) The fifth scenario (Scenario 5, S5) adds two dynamic obstacles to the training environment. These two obstacles are circular with a radius of 3 m and both move at a speed of 0.5 m/s.

We randomly tested each scenario 500 times with the same random seed as reported in Section IV-B. The results are listed in Table IV. As shown in Table IV, all five policies achieved

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

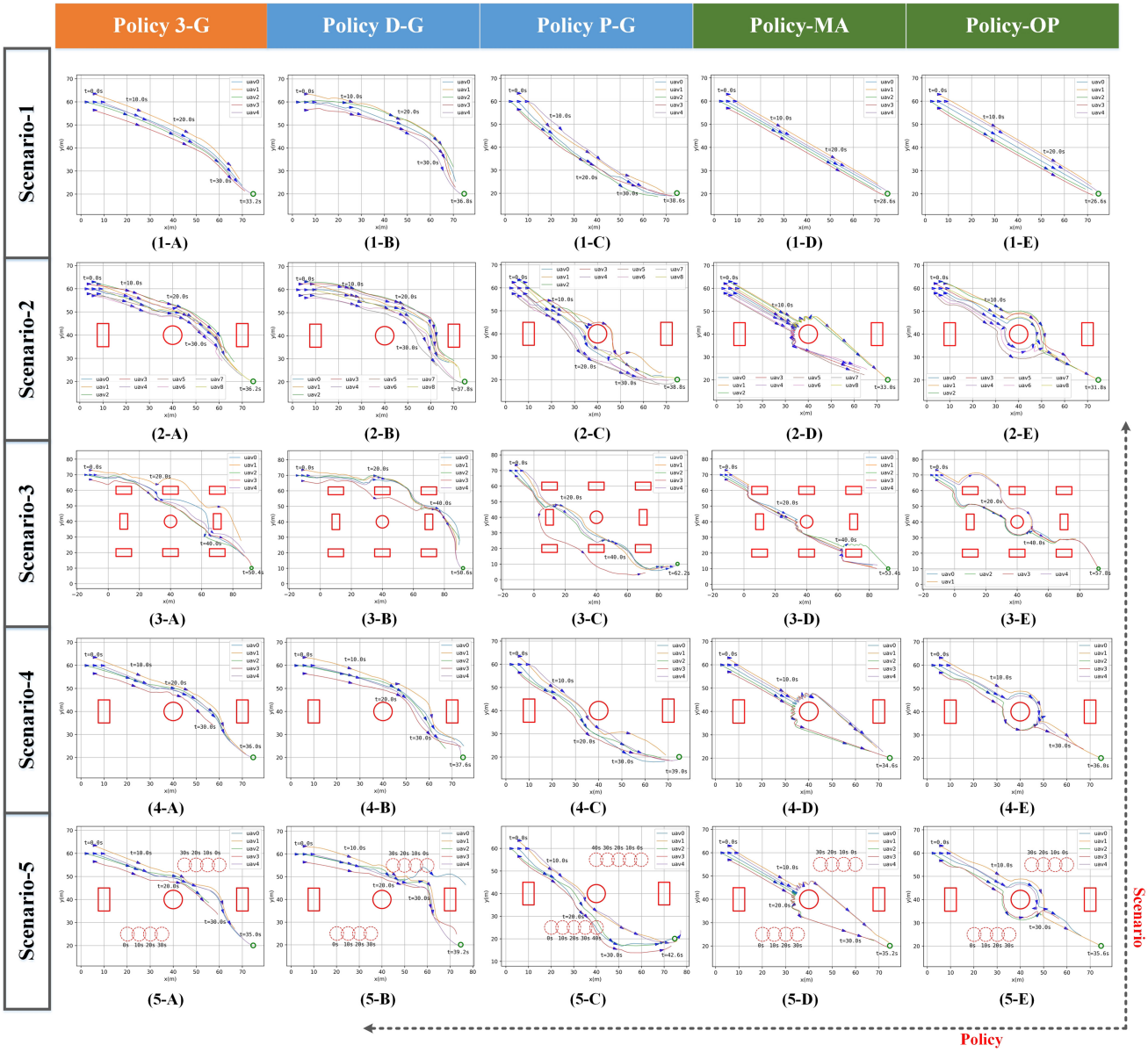

Fig. 6. Generalization tests results.

TABLE V
GENERALIZATION TEST RESULTS OF POLICY3-G UNDER DIFFERENT VALUES OF τ_v IN THE TRAINING SCENARIO

	$d_{\min}^{\text{obs}}$(m)	$d_{\min}^{\text{uav}}$(m)	d_c(m)	V_{error}^c(m/s)	φ_{error}^c(rad)	V(m/s)	t^g(s)	Success rate
$\tau_v = 0.2$s	3.357±0.666	**1.802±0.105**	3.426±0.337	0.311±0.056	0.149±0.181	**2.102±0.064**	**35.348±0.729**	87.0%
$\tau_v = 0.5$s	3.082±0.770	1.627±0.219	3.652±0.650	0.315±0.060	0.178±0.234	2.072±0.069	35.797±0.850	93.4%
$\tau_v = 1.0$s	**3.549±0.937**	1.627±0.276	**3.393±0.610**	0.268±0.067	0.159±0.219	2.052±0.076	36.709±1.434	**93.8%**
$\tau_v = 1.5$s	3.329±0.786	1.702±0.222	3.650±0.660	0.263±0.057	0.153±0.207	2.028±0.066	36.743±1.250	93.6%
$\tau_v = 2.0$s	3.359±0.807	1.678±0.236	3.634±0.678	**0.242±0.055**	**0.147±0.191**	2.000±0.061	37.574±1.908	92.6%

a success rate of 100% in Scenario 1. In particular, both Policy-MA and Policy-OP outperformed learning-based methods, indicating that traditional methods can effectively control the UAV swarm in free environments to form more compact and consistent flocks compared to the proposed learning-based methods, as demonstrated by the low values of d_c, V_{error}^c, and φ_{error}^c. It is worth noting that Policy3-G outperformed PolicyD-G and PolicyP-G in all metrics. Fig. 6(1) illustrates the trajectories of the UAVs in Scenario 1. It can be observed that UAVs controlled by Policy-MA and Policy-OP have a stable flock topology and shorter flight times, and Policy-OP achieves particularly impressive results by maintaining the initial flock topology throughout the entire flight. On the contrary, UAVs controlled by learning-based methods have a longer flight time and a dynamically changing flock topology, indicating that traditional methods are more suitable for flocking control of UAV swarms in free environments.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

In Scenarios 2 and 3, the success rates of all policies decrease. As the number of UAVs and obstacles increases, it becomes more challenging for UAVs to maintain control in a flock. Compared to Policy3-G, both the learning-based and traditional methods do not perform as well and have low success rates; in particular, in Scenario 2, PolicyP-G has a success rate of less than 70%, and in Scenario 3, PolicyP-G, Policy-MA, and Policy-OP have success rates of less than 80%. The limitations of these two traditional policies are highlighted in an environment of dense obstacles. Fig. 6(2) shows the trajectories of UAVs in Scenario 2. As shown, Policy3-G and PolicyD-G are able to guide the nine UAVs to form a flock while navigating around obstacles. However, PolicyP-G controls the UAVs too closely to the central obstacle, resulting in quick collisions and thus a low success rate. Furthermore, the UAVs controlled by Policy-MA and Policy-OP are split into two groups to avoid collisions with obstacles, contributing to the high values of d_c, V^c_{error}, and φ^c_{error}, as listed in Table IV.

Furthermore, in Scenario 3, as the number of obstacles increases, the UAV swarm is divided into multiple flocks, as shown in Fig. 6(3). Fig. 6(3-A) illustrates that the UAV swarm under Policy3-G is divided into two groups to avoid obstacles in $t = 20$ s, which contributes to the increase in the value of d_c. In contrast to Fig. 5(k) and Fig. 6(1-D), the UAV swarm controlled by Policy-MA avoids obstacles together without dividing into multiple parts. As seen in Table IV, in Scenario 3, the minimum distance $d^{\text{uav}}_{\text{min}}$ between the UAVs controlled by Policy-MA is smaller, which is due to Policy-MA's inability to balance inter-UAV collision avoidance and obstacle collision avoidance. The results in Fig. 6(3-E) are consistent with the results in Fig. 5(l) and Fig. 6(2-E). It is worth noting that the increase in d_{env} leads to an increase in t^g in Scenario 3.

Compared to the results in Table III, the results in Scenario 4 listed in Table IV show that UAV observation errors negatively impact UAV flocking control performance. The success rates of all policies are reduced. However, Policy3-G still has a success rate greater than 90%, indicating that Policy3-G has better robustness to UAV observation errors than other methods. Fig. 6(4) illustrates the trajectories of UAVs in Scenario 4, which are similar to the UAV trajectories in Fig. 5, indicating that the UAV observation errors have little influence on the UAV trajectories.

From the results of Scenario 5, we found that dynamic obstacles significantly impact the control of the UAV flocking. First, dynamic obstacles result in smaller distances between UAVs and obstacles, as well as other UAVs, that is, smaller values of $d^{\text{obs}}_{\text{min}}$ and $d^{\text{uav}}_{\text{min}}$, resulting in a lower success rate. For the learning-based methods, environments with dynamic obstacles are not encountered during the training and, therefore, cannot handle this situation well. Policy-MA and Policy-OP have poor obstacle avoidance capabilities, and dynamic obstacles increase obstacle avoidance difficulty. Second, to avoid obstacles, the compactness and consistency of the UAV flock become worse, and the values of d_c, V^c_{error}, and φ^c_{error} all increase. In comparison, Policy3-G has a success rate of more than 80% and better-flocking control metrics. Fig. 6(5) illustrates the trajectories of the UAVs and the locations of the dynamic obstacles at different times. As seen under policies Policy3-G, PolicyD-G, and PolicyP-G, the trajectories of the UAVs are all affected by dynamic obstacles and the flocks are more dispersed. In contrast, under policies Policy-MA and Policy-OP, the trajectories of the UAVs are unchanged because the dynamic obstacles are farther away from the UAVs.

The results of our tests demonstrate that our proposed flocking control policy, Policy3-G, has a good generalization ability in environments not encountered during training, indicating that it has strong robustness to changes in the external environment. We also tested the generalization ability of Policy3-G to changes in the UAV model. As shown in (1), the motion performance of the UAV is mainly determined by the parameter τ_v. Therefore, we tested the flocking control performance of Policy3-G with different values of τ_v. The results, listed in Table V, show that when the value of τ_v differs from the training environment ($\tau_v = 1.0$ s), the success rates decrease, with the lowest success rate of 87% observed when $\tau_v = 0.2$ s. This decrease in success rate is due to the fact that when $\tau_v < 1.0$ s, the response to the UAV velocity command increases, leading to an increase in average speed and a higher likelihood of collisions. Similarly, when $\tau_v > 1.0$ s, the velocity command response capability of the UAV decreases, leading to a decrease in average speed and a higher likelihood of collisions. However, in general, when the parameter τ_v changes, Policy3-G can enable the UAV swarm to maintain a high success rate (at least 87%) and high flocking control performance, indicating that it also has good generalizability to changes in the UAV model.

Although a large number of simulation results demonstrate that our proposed DRL-based flocking controller has a good generalization ability to changes in the external environment and the internal UAV model, it is not trivial to theoretically analyze the stability of the flocking controller. On the one hand, as the flocking controller is represented as a DNN with a complex structure and many parameters, it is difficult to effectively analyze its input–output response characteristics. Although the stability of neural network-based controllers with a single hidden layer has been analyzed in [19] and [38], it remains challenging to analyze the stability of neural network-based controllers with multiple hidden layers. On the other hand, since DRL is a model-free approach, there is a lack of suitable UAV models to analyze the stability of the DRL-based flocking controller. In conclusion, the theoretical analysis of the stability of DRL-based controllers poses a significant challenge and is a key factor that limits the practical application of DRL-based controllers.

V. Conclusion

In this study, we addressed the problem of flocking control of a UAV swarm in continuous state and action spaces. First, we formulated the flocking control problem in an RL framework and solved it using an actor-critic DRL method, SAC. Subsequently, the inter-UAV collision avoidance capability was considered an instinct of the UAVs and was implemented using a repulsion function. Furthermore, we investigated communication denial environments, where UAVs used their onboard sensors to perceive the states of other UAVs.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.

The flocking control policy was trained through a CTDE paradigm, where the UAVs' experience was used to train the shared policy network. Specifically, we used a centralized critic augmented with additional information on the entire UAV swarm to facilitate training, which positively affected the policy, particularly when UAVs have limited perception capacity. Finally, we conducted simulation experiments to verify the performance of the proposed algorithm. From the results, we can draw the following conclusions.

1) The policy with the repulsion function and the limited visual field exhibited a high success rate of up to 93.8% in training environments.
2) The repulsion function and the limited visual field caused the UAVs to flock more sparsely at high speed.
3) The policy with the repulsion function and the limited visual field exhibited robust generalization performance: an 85.6% success rate in environments with a high number of UAVs and a 91.2% success rate in environments with a high number of obstacles and an 82.2% success rate in environments with dynamic obstacles.
4) Traditional methods are more suitable for flocking control of UAV swarms in free environments, whereas the proposed learning-based methods are more suitable in cluttered environments.

In future research, we will further establish a more accurate 6DOF model of the UAV and extend the motion process of the UAV to 3-D space. In addition, we will further validate our proposed flocking control policy on real UAV platforms to improve its performance in real flights and continue to focus on the theoretical stability analysis of the proposed DRL-based controller.

REFERENCES

[1] J. Wu, C. Luo, Y. Luo, and K. Li, "Distributed UAV swarm formation and collision avoidance strategies over fixed and switching topologies," *IEEE Trans. Cybern.*, vol. 52, no. 10, pp. 10969–10979, Oct. 2022.

[2] A. Bejaoui, K.-H. Park, and M.-S. Alouini, "A QoS-oriented trajectory optimization in swarming unmanned-aerial-vehicles communications," *IEEE Wireless Commun. Lett.*, vol. 9, no. 6, pp. 791–794, Jun. 2020.

[3] J. Tian, W. Tao, W. Weiping, L. Xiaobo, and Z. Xin, "An adaptive scale control method of multiple UAVs for persistent surveillance," *J. Comput. Res. Develop.*, vol. 55, no. 6, p. 1254, 2018.

[4] S. Liu, K. Mohta, S. Shen, and V. Kumar, "Towards collaborative mapping and exploration using multiple micro aerial robots," in *Proc. 14th Int. Symp. Exp. Robot.*, 2016, pp. 865–878.

[5] C. Ju and H. I. Son, "A distributed swarm control for an agricultural multiple unmanned aerial vehicle system," *Proc. Inst. Mech. Eng. I, J. Syst. Control Eng.*, vol. 233, no. 10, pp. 1298–1308, 2019.

[6] H. Fang, Y. Wei, J. Chen, and B. Xin, "Flocking of second-order multiagent systems with connectivity preservation based on algebraic connectivity estimation," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1067–1077, Apr. 2017.

[7] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, "Optimized flocking of autonomous drones in confined environments," *Sci. Robot.*, vol. 3, no. 20, 2018, Art. no. eaat3536.

[8] H. Qiu and H. Duan, "A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles," *Inf. Sci.*, vol. 509, pp. 515–529, Jan. 2020.

[9] Y. Lyu, J. Hu, B. M. Chen, C. Zhao, and Q. Pan, "Multivehicle flocking with collision avoidance via distributed model predictive control," *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2651–2662, May 2021.

[10] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proc. 14th Annu. Conf. Comput. Graph. Interact. Techn.*, 1987, pp. 25–34.

[11] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 401–420, Mar. 2006.

[12] W. Liu and Z. Gao, "A distributed flocking control strategy for UAV groups," *Comput. Commun.*, vol. 153, pp. 95–101, Mar. 2020.

[13] D. Sun, C. Kwon, and I. Hwang, "Hybrid flocking control algorithm for fixed-wing aircraft," *J. Guid., Control, Dyn.*, vol. 42, no. 11, pp. 2443–2455, 2019.

[14] K. Guo, X. Li, and L. Xie, "Ultra-wideband and odometry-based cooperative relative localization with application to multi-UAV formation control," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2590–2603, Jun. 2020.

[15] D. B. Wilson, A. H. Göktoğan, and S. Sukkarieh, "Vision-aided guidance and navigation for close formation flight," *J. Field Robot.*, vol. 33, no. 5, pp. 661–686, 2016.

[16] Y. Tang et al., "Vision-aided multi-UAV autonomous flocking in GPS-denied environment," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 616–626, Jan. 2019.

[17] L. Zheng and Z. Zhang, "Convergence and robustness analysis of novel adaptive multilayer neural dynamics-based controllers of multirotor UAVs," *IEEE Trans. Cybern.*, vol. 51, no. 7, pp. 3710–3723, Jul. 2021.

[18] M. Wang, B. Chen, and C. Lin, "Prescribed finite-time adaptive neural trajectory tracking control of quadrotor via output feedback," *Neurocomputing*, vol. 458, pp. 364–375, Oct. 2021.

[19] O. Elhaki and K. Shojaei, "A novel model-free robust saturated reinforcement learning-based controller for quadrotors guaranteeing prescribed transient and steady state performance," *Aerosp. Sci. Technol.*, vol. 119, Dec. 2021, Art. no. 107128.

[20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[21] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[22] W. Wang and X. Chen, "Model-free optimal containment control of multi-agent systems based on actor-critic framework," *Neurocomputing*, vol. 314, pp. 242–250, Nov. 2018.

[23] X. Zhang, Y. Liu, X. Xu, Q. Huang, H. Mao, and A. Carie, "Structural relational inference actor-critic for multi-agent reinforcement learning," *Neurocomputing*, vol. 459, pp. 383–394, Oct. 2021.

[24] H. M. La, R. Lim, and W. Sheng, "Multirobot cooperative learning for predator avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 52–63, Jan. 2015.

[25] S.-M. Hung and S. N. Givigi, "A Q-learning approach to flocking with UAVs in a stochastic environment," *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 186–197, Jan. 2017.

[26] C. Yan, X. Xiang, and C. Wang, "Fixed-wing UAVs flocking in continuous spaces: A deep reinforcement learning approach," *Robot. Auton. Syst.*, vol. 131, Sep. 2020, Art. no. 103594.

[27] C. Yan, C. Wang, X. Xiang, Z. Lan, and Y. Jiang, "Deep reinforcement learning of collision-free flocking policies for multiple fixed-wing UAVs using local situation maps," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1260–1270, Feb. 2022.

[28] Z. Xu, Y. Lyu, Q. Pan, J. Hu, C. Zhao, and S. Liu, "Multi-vehicle flocking control with deep deterministic policy gradient method," in *Proc. IEEE 14th Int. Conf. Control Autom. (ICCA)*, 2018, pp. 306–311.

[29] C. Wang, J. Wang, and X. Zhang, "A deep reinforcement learning approach to flocking and navigation of UAVs in large-scale complex environments," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, 2018, pp. 1228–1232.

[30] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6382–6393.

[31] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2017, pp. 66–83.

[32] T. Haarnoja et al., "Soft actor-critic algorithms and applications," 2018, *arXiv:1812.05905*.

[33] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[34] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.

[36] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–10.

[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[38] O. Elhaki and K. Shojaei, "Output-feedback robust saturated actor–critic multi-layer neural network controller for multi-body electrically driven tractors with n-trailer guaranteeing prescribed output constraints," *Robot. Auton. Syst.*, vol. 154, Aug. 2022, Art. no. 104106.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 03,2024 at 11:24:22 UTC from IEEE Xplore. Restrictions apply.