Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A multi-agent collaboration scheme for energy-efficient task scheduling in a 3D UAV-MEC space*#

Yang LI[1], Ziling WEI‡[1], Jinshu SU‡[1,2], Baokang ZHAO[1]

[1]*College of Computer, National University of Defense Technology, Changsha 410073, China*

[2]*Academy of Military Sciences, Beijing 100091, China*

E-mail: liyang20@nudt.edu.cn; weiziling@nudt.edu.cn; sjs@nudt.edu.cn; bkzhao@nudt.edu.cn

Received June 1, 2023; Revision accepted Nov. 13, 2023; Crosschecked May 27, 2024

Abstract: Multi-access edge computing (MEC) presents computing services at the edge of networks to address the enormous processing requirements of intelligent applications. Due to the maneuverability of unmanned aerial vehicles (UAVs), they can be used as temporal aerial edge nodes for providing edge services to ground users in MEC. However, MEC environment is usually dynamic and complicated. It is a challenge for multiple UAVs to select appropriate service strategies. Besides, most of existing works study UAV-MEC with the assumption that the flight heights of UAVs are fixed; i.e., the flying is considered to occur with reference to a two-dimensional plane, which neglects the importance of the height. In this paper, with consideration of the co-channel interference, an optimization problem of energy efficiency is investigated to maximize the number of fulfilled tasks, where multiple UAVs in a three-dimensional space collaboratively fulfill the task computation of ground users. In the formulated problem, we try to obtain the optimal flight and sub-channel selection strategies for UAVs and schedule strategies for tasks. Based on the multi-agent deep deterministic policy gradient (MADDPG) algorithm, we propose a curiosity-driven and twin-networks-structured MADDPG (CTMADDPG) algorithm to solve the formulated problem. It uses the inner reward to facilitate the state exploration of agents, avoiding convergence at the sub-optimal strategy. Furthermore, we adopt the twin critic networks for update stabilization to reduce the probability of Q value overestimation. The simulation results show that CTMADDPG is outstanding in maximizing the energy efficiency of the whole system and outperforms the other benchmarks.

Key words: Multi-access edge computing; Multi-agent reinforcement learning; Unmanned aerial vehicles; Task scheduling

https://doi.org/10.1631/FITEE.2300393 **CLC number:** TP301.6

1 Introduction

The increasingly up-to-date information technology has accelerated the development of various intelligent applications, and these applications aim at the successful execution of numerous tasks together with a stringent consideration of needing to minimize the consumption of resources, including time, thereby achieving an effective delay minimization. In such a circumstance, the resultant computation demands become challenging. If all tasks are delivered to the cloud center, it will induce the explosive growth of network traffic, which imposes a huge burden on the backbone network. In addition, the latency of task data transmission is high since the tasks are generally far from the cloud server. In this case, multi-access edge computing (MEC) (Ding

‡ Corresponding authors

* Project supported by the National Natural Science Foundation of China (Nos. 62202486 and U22B2005)

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2300393) contains supplementary materials, which are available to authorized users

ORCID: Yang LI, https://orcid.org/0000-0002-5152-1178; Ziling WEI, https://orcid.org/0000-0002-7858-1445; Jinshu SU, https://orcid.org/0000-0001-9273-616X

et al., 2023) is an excellent choice for the provision of real-time processing services. It extends cloud computing by providing computing services near the data sources. As for the deployment of edge servers, they can be deployed at fixed locations, such as base station (BS). Nevertheless, once BS malfunctions occur, it leads to paralysis of the edge network.

As an auxiliary edge server, unmanned aerial vehicles (UAVs) can be deployed to offer the computational resources on demand. The combination of UAV and MEC, called UAV-MEC, provides high-quality edge services with flexibility and maneuverability characteristics. In UAV-MEC, the UAV equipped with a communication module serves as an aerial BS to communicate with ground users. It processes the tasks from these users or merely acts as a communication relay to forward the task data. The UAVs' service environment is quite complicated and volatile. There are various densities of user clusters and different tasks under diverse geographical conditions. Manually navigating the UAVs with suitable flight and schedule strategies is extremely difficult. The inappropriate strategies consume much energy and increase the task computation delay. Meanwhile, the UAV's energy is a crucial resource, which directly determines the endurance of flight. The flight strategy refers to flying deployment guidance for a UAV during its cruise, related to direction and speed. The schedule strategy represents a scheme to properly schedule various tasks to be processed on different edge servers with diverse resource allocation strategies, including UAVs and BSs.

Therefore, it is necessary to develop an intelligent module for UAVs to adjust their flight and schedule strategies automatically. There have been many works maximizing the edge server's utility via optimization of both UAV deployment and resource allocation in UAV-MEC scenarios. Some works (Zhou et al., 2018; Ashraf Ateya et al., 2019; Liu JF et al., 2019) dispatch UAVs around BSs for an auxiliary assist, aiming at facilitating the edge service. However, the locations of the dispatched UAVs cannot be dynamically adjusted with the change of edge environment. Other works (Jiang et al., 2020; Wang JR et al., 2020; Zhang and Ansari, 2020) focus on the flexible flight trajectory adjustment during the UAV's cruise. For simplicity, they usually assumed that the UAV's flying height is fixed. It indicates that the UAV flies in a two-dimensional (2D) plane rather than a three-dimensional (3D) space. Nevertheless, higher flying height means a higher probability of establishing a line of sight (LoS) link between the UAVs and ground users (Liao et al., 2021). Furthermore, according to Al-Hourani et al. (2014), the radio coverage scope will increase with a higher flying height of a UAV within 2 km. Hence, the flying height is a significant element in the edge communication environment. Nevertheless, most of existing works omit the impact of flying height. Our approach, on the other hand, is to fully exploit the significance of the concept of flying height in the edge communication environment, so that a study of the UAV-MEC scenario that aligns with reality is facilitated.

In this paper, we study the flying and serving process of multiple UAVs in a 3D space, where the UAVs can dynamically adjust their flying heights. Through the UAVs' flight deployment and the task scheduling, we focus on an optimization problem aiming at maximizing energy efficiency with consideration of the co-channel interference. To solve the optimization problem, it is formulated as a multi-agent extension of Markov decision processes (MDPs) in deep reinforcement learning (DRL) (Liu XY et al., 2022).

The main contributions of this paper are summarized as follows:

1. We present a novel UAV-MEC scenario, which aligns with reality. In this scenario, multiple UAVs dynamically fly in a 3D space to provide edge services to ground users under the circumstance of co-channel interference. During the cruise of UAVs, they flexibly adjust the flying action, select the sub-channel for communication, and generate proper task scheduling strategies. These decisions are fully comprehensive, based on which the optimization problem of energy efficiency is studied to improve the overall performance.

2. We propose a multi-agent reinforcement learning algorithm to facilitate the state exploration and optimize the overall energy efficiency cooperatively. Via co-innovation, all agents have cooperation to avoid potential collisions and coordinate their decisions. Under such circumstances, all isolated information is integrated to support the optimal strategy. Each collaborative agent unselfishly makes its strategy to maximize the overall reward. To fully explore the environment state, a discounted intrinsic reward is considered, motivating the curiosity of the

agent. Furthermore, we adopt twin critic networks for stable update to tackle the instability problem of Q value in the critic networks. Hence, each UAV can dynamically adjust its flying strategy and its serving task scheduling strategies to maximize the number of fulfilled tasks with the minimum energy consumption.

3. We conduct abundant simulations to evaluate the performance of the proposed algorithm. The results have proved that our proposed algorithm is outstanding in improving the overall system performance. The training reward shows at least a 19% improvement compared to the baseline multi-agent deep deterministic policy gradient (MADDPG) algorithm.

2 Related works

To efficiently implement the specific optimization goals during task scheduling, many works adopt the DRL method to improve the system performance. This section reviews the related works, where the UAV flying in a 3D space acts as a DRL agent to make intelligent decisions. In Liu Q et al. (2020), a single UAV was served as an agent in DRL. To ensure the quality of service (QoS) of each terminal user, the UAV with limited energy dynamically plans its trajectory according to the locations of mobile terminal users, based on the double deep Q networks (DQNs). However, a single UAV's coverage scope is limited. Its computation resource cannot support large-scale computing. Hence, the MEC scenario with a single UAV providing computation service is not functional in actual situations.

Inversely, multiple UAVs indicate a more enormous communication coverage scope and more available resources compared with one UAV, and are efficient in complex environments. The multi-agent algorithm possesses the features of autonomy, coordination, and self-organization. It plays the role of conductor, which automatically coordinates different learning agents to make mutual effort, further maximizing the overall performance. The works (Dai C et al., 2022; Dai ZJ et al., 2022; Wang LY et al., 2022; Wang ZQ et al., 2022; Wu et al., 2022; Xia et al., 2022; Xu S et al., 2022; Yin et al., 2022; Zhao et al., 2022; Zhong et al., 2022; Lakew et al., 2023; Wang JZ et al., 2023) use multiple UAVs to serve as collaborative agents. In Wang ZQ et al. (2022), UAVs were used as mobile BSs to offer MEC services for user ends. The authors proposed a multi-agent path planning scheme based on DRL. Besides, fairness was considered to ensure the user end offloading balance and the UAV load balance. In Dai ZJ et al. (2022), a resource allocation algorithm for the UAV network based on multi-agent collaborative environment learning was proposed. Each UAV was modeled as an independent agent, which improves the UAV network's utility through the dynamic selection decisions of its deployment position, transmission power, and occupied sub-channels. Based on the K-means algorithm and the DRL in a distributed manner and a centralized manner (Xu S et al., 2022), the mission time was minimized with constraints of UAV's maximum speed and acceleration, collision avoidance, and communication interference among UAVs. A centralized multi-agent Q-learning algorithm was built, with which multiple UAVs updated their positions in a joint manner in Wang LY et al. (2022). The uplink transmission in a UAV-assisted cellular network was studied to minimize the transmission power consumption of users and UAVs through designing the proper UAV deployment and association schemes. In Wu et al. (2022), UAVs and unmanned ground vehicles were deployed as BSs, where a federated MADDPG based trajectory optimization algorithm was proposed to maximize the average spectrum efficiency.

Even though existing DRL methods are effective in task scheduling, their performance remains to be improved. Many multi-agent DRL algorithms suffer from inherent disadvantages. Regarding the DQN-based or deep deterministic policy gradient (DDPG) based multi-agent DRL algorithms, Q value overestimation is a critical problem that may severely degrade the performance. Apart from this, the action exploration is usually prematurely converged to a sub-optimal status in the multi-agent environment. Therefore, we aim to solve the mentioned problems to improve the DRL algorithm performance. Besides, most of existing works (Liu Q et al., 2020; Yang et al., 2020; Yu Z et al., 2020; Ji et al., 2021; Joo et al., 2021; Tun et al., 2021; Wang L et al., 2021; Xu Y et al., 2021) assume that the UAV flies at a fixed height, and that its flying actions are essentially 2D, not 3D. They ignored the impact of the flying height, which influences the coverage scope and communication delay. Although the works (Mei

et al., 2020; Liao et al., 2021; Zhong et al., 2022) allow UAVs to individually fly in a height scope, their values ([50 m, 300 m], [20 m, 150 m], and [50 m, 150 m]) are too small to seek out the optimal UAV deployment location in reality. To sufficiently investigate our scenario in a 3D space, we research the optimal flying height within the range of [10 m, 1000 m]. Our work tries to solve the energy efficiency optimization problem via a multi-agent DRL algorithm in a 3D flight space. To achieve the optimization goal, the trajectory design, task scheduling, and sub-channel selection are involved in considering the flying energy consumption.

3 System model

In this section, we consider the issue of maximizing the total energy efficiency of UAVs. As shown in Fig. 1, the UAV-MEC scenario comprises N UAVs, U users, and one remote BS. The sets of UAVs and users are defined as $\mathcal{N}=\{1,2,...,N\}$ and $\mathcal{U}=\{1,2,...,U\}$, respectively. The runtime of UAVs concerning the provision of edge services is divided into T time slots, i.e., $\mathcal{T}=\{1,2,...,T\}$. The duration of each time slot $t\in\mathcal{T}$ is represented as τ. In this scenario, UAV n serves multiple users, and user u connects only to one UAV for edge service. An association relationship between user u and UAV n is represented as the binary variable $a^t_{u,n}$. If $a^t_{u,n}=1$, user u is associated with UAV n. Otherwise, user u cannot send its task data to UAV n for computation. Thus, we have $\sum_{n=1}^{N}a^t_{u,n}\le 1, \forall u\in\mathcal{U}$. It is assumed that the near BS is malfunctioning, and that UAVs are deployed to help ground users process their tasks as temporal BSs. Since the remote BS e is running normally, UAVs can decide whether to relay the task data to e for efficient computation. $b^t_{u,e}$ denotes the computation relationship between user u and temporal BS e, determined by UAV n. If $b^t_{u,e}=1$, the task data of user u will be transmitted to e. In contrast, $b^t_{u,e}=0$ denotes that the task will be locally processed on UAV n. The orthogonal multiplexing technology is adopted in the wireless communication process. The available bandwidth is divided into K sub-channels, denoted as $\mathcal{K}=\{1,2,...,K\}$. K is subjected to $K<N$, which means that at least two UAVs occupy the same sub-channel (i.e., co-channel). Thus, there exists interference between UAVs, and the communication time would increase. In Fig. 1, UAVs n_i and n_{i-1} have co-channel interference. Let binary variable $c^t_{n,k}$ indicate the sub-channel utilization relationship for a UAV. If UAV n occupies sub-channel k, then $c^t_{n,k}=1$, and $c^t_{n,k}=0$ otherwise. Each UAV can use only one sub-channel, and thus $\sum_{k=1}^{K}c^t_{n,k}=1, \forall n\in\mathcal{N}$. Since the result of a task is much smaller than its original data, the download time is assumed to be negligible. Similar to the approaches adopted in the studies (Gu et al., 2021; Joo et al., 2021; Yu Y et al., 2021), we consider this a reasonable assumption to make in MEC. Besides, UAVs can flexibly adjust their flying height during a cruise in a 3D space, which is more intricate than that in a 2D plane. A detailed description of notations employed in this paper is provided in Table S1 in the supplementary materials.

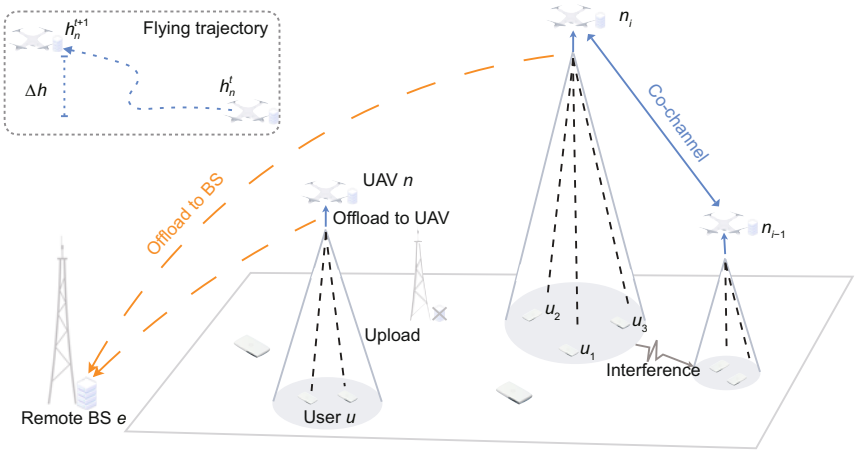

Fig. 1 Schedule scenario of UAVs' flight and users' tasks in a 3D space

3.1 Task model

In UAV-MEC, UAVs are responsible for processing the tasks from users. Each user's task data are randomly generated during different time slots. A user's task is represented as

$$\text{task}^t(u) = \{D_u^t, C_u^t, \bar{D}_u^t\},\ \forall u \in \mathcal{U}, \tag{1}$$

where D_u^t is the task data size, C_u^t denotes the number of required cycles of the central processing unit (CPU) per bit, and $\bar{D}_u^t$ represents the task deadline. To specify the correlation of D_u^t and $\bar{D}_u^t$, $\bar{D}_u^t$ is calculated as

$$\bar{D}_u^t = \frac{D_u^t}{\max(D_{\mathcal{U}})}\max(\bar{D}_{\mathcal{U}}), \tag{2}$$

where $\max(D_{\mathcal{U}})$ is the global maximum value of the data size and $\max(\bar{D}_{\mathcal{U}})$ is the global maximum value of the deadline. The maximum values are pre-configured. In our work, the task data size D_u^t is randomly generated from the range of $[\min(D_{\mathcal{U}}), \max(D_{\mathcal{U}})]$, where $\min(D_{\mathcal{U}})$ is the global minimum value of the data size. In general, when a task data size is large, its deadline should be increased to reserve adequate time for computation. As a consequence, in Eq. (2), we use the proportion of data size $\frac{D_u^t}{\max(D_{\mathcal{U}})}$ to determine the corresponding deadline, which is derived by multiplying the proportion with the global maximum deadline $\max(\bar{D}_{\mathcal{U}})$.

3.2 Mobility model

In time slot t, the 3D coordinates of UAV n and user u are $\text{loc}_n^t(3\text{D}) = (x_n^t, y_n^t, h_n^t)$ and $\text{loc}_u^t(3\text{D}) = (x_u^t, y_u^t, 0)$, respectively, where $X_{\min} \le x_n^t \le X_{\max}$, $Y_{\min} \le y_n^t \le Y_{\max}$, and $H_{\min} \le h_n^t \le H_{\max}$. Here, x, y, and h refer to the points in X-, Y-, and H-axis, whose minimum values individually are $X_{\min}$, $Y_{\min}$, and $H_{\min}$, and correspondingly, their maximum values are $X_{\max}$, $Y_{\max}$, and $H_{\max}$, respectively. For ease of calculating the X-Y axis distance, their 2D coordinates are involved and represented as $\text{loc}_n^t(2\text{D}) = (x_n^t, y_n^t)$ and $\text{loc}_u^t(2\text{D}) = (x_u^t, y_u^t)$. According to Al-Hourani et al. (2014), within the flying height range of [0 m, 2000 m], the coverage radius shows a positive growth trend, which is nearly in direct proportion to the flying height. The coefficient of direct proportion is approximately equal to 1. Therefore, for simplicity, the coverage radius is determined by the flying height within the range of [10 m, 1000 m] in our work, prevailing a numerical equality between the coverage radius and the flying height range. The radio coverage scope of a UAV will increase with a higher flying height. If user u wants to offload data to UAV n in t, it must be inside the radio coverage scope of UAV n, represented as r_n^t. The location of user u needs to meet the constraint $\|\text{loc}_n^t(2\text{D}) - \text{loc}_u^t(2\text{D})\|_2 \le r_n^t$.

3.3 Communication model

Since we consider only the upload communication process, the communication model can be divided into two parts: (1) from a user to a UAV; (2) from a UAV to the remote BS.

3.3.1 Communication from a user to a UAV

The probabilistic air-to-ground model (Liao et al., 2021; Dai C et al., 2022; Wang LY et al., 2022) is adopted to formulate the average path loss $\text{PL}^t(u,n)$ during the wireless communication between user u and UAV n. The formulation is

$$\text{PL}^t(u,n) = P_{\text{LoS}}^t(u,n)L_{\text{LoS}}^t + P_{\text{NLoS}}^t(u,n)L_{\text{NLoS}}^t, \tag{3}$$

where L_{LoS}^t and L_{NLoS}^t represent the path loss in LoS and non-line of sight (NLoS) links, respectively. They are given as

$$L_{\text{LoS}}^t = \left(\frac{4\pi f_{\text{c}}}{c}\right)^2 \left(d^t(u,n)\right)^2 \eta_{\text{LoS}}, \tag{4}$$

$$L_{\text{NLoS}}^t = \left(\frac{4\pi f_{\text{c}}}{c}\right)^2 \left(d^t(u,n)\right)^2 \eta_{\text{NLoS}}, \tag{5}$$

where f_{c} represents the carrier frequency, c is the light velocity, $d^t(u,n) = \|\text{loc}_u^t(3\text{D}) - \text{loc}_n^t(3\text{D})\|_2$ denotes the straight-line distance between user u and UAV n, and η_{LoS} and η_{NLoS} indicate the average additional loss for LoS and NLoS links, respectively. In Eq. (3), the probability of LoS link $P_{\text{LoS}}^t(u,n)$ between user u and UAV n is represented as

$$P_{\text{LoS}}^t(u,n) = \frac{1}{1 + c_1 \exp\left(-c_2\left(\frac{180}{\pi}\bar{\theta}_{u,n}^t - c_1\right)\right)}, \tag{6}$$

where c_1 and c_2 are constant values of the communication environment, and $\bar{\theta}_{u,n}^t$ is the elevation angle of user u to UAV n, calculated by $\bar{\theta}_{u,n}^t = \arcsin\left(\frac{h_n^t}{d^t(u,n)}\right)$. With $P_{\text{LoS}}^t(u,n)$, we can derive $P_{\text{NLoS}}^t(u,n) = 1 - P_{\text{LoS}}^t(u,n)$.

As mentioned, the co-channel interference is involved in the UAV-MEC scenario. It is necessary to calculate the interference among the UAVs that share the same sub-channel. If user u's associated UAV n occupies sub-channel k, the co-channel interference from other users served by other UAVs is depicted as

$$I_k^t(u',n)=\sum_{\substack{u'\in\mathcal{U}\setminus\{u\}\\ n'\in\mathcal{N}\setminus\{n\}}} a_{u',n'}^t c_{n',k}^t p_{u'}\mathrm{PL}^t(u',n), \quad (7)$$

where $p_{u'}$ represents the transmission power of user u'. For convenience, the transmission powers of all users are set to be the same value. The signal-to-interference-plus-noise ratio (SINR) is derived as

$$\mathrm{SINR}_k^t(u,n)=\frac{a_{u,n}^t c_{n,k}^t p_u \mathrm{PL}^t(u,n)}{I_k^t(u',n)+\Theta}, \quad (8)$$

where p_u represents the transmission power of user u, and Θ is the noise power.

According to the Shannon capacity formula, the upload transmission rate of user u is written as

$$r_k^t(u,n)=B_1\log_2\left(1+\mathrm{SINR}_k^t(u,n)\right), \quad (9)$$

where B_1 is the uplink bandwidth of user u. The transmission time in the uplink is calculated as

$$T_{\mathrm{tran}}^t(u,n)=\frac{D_u^t}{r_k^t(u,n)}. \quad (10)$$

3.3.2 Communication from a UAV to the remote BS

As aforementioned, UAV n needs to transmit user u's task data to the remote BS e when $b_{u,e}^t=1$. The free space path loss model is used to formulate the communication process from a UAV to the remote BS for convenience. The channel gain is derived by

$$G^t(n,e)=\left(\frac{4\pi f_{\mathrm{c}}}{c}d^t(n,e)\right)^2, \quad (11)$$

where $d^t(n,e)$ is the distance between UAV n and the remote BS e, which is calculated as $d^t(n,e)=\|\mathrm{loc}_n^t(3\mathrm{D})-\mathrm{loc}_e^t(3\mathrm{D})\|_2$. The SINR is denoted as

$$\mathrm{SINR}^t(n,e)=\frac{b_{n,e}^t p_n G^t(n,e)}{\Theta}, \quad (12)$$

where p_n denotes the transmission power of UAV n, and $b_{n,e}^t\in\{0,1\}$ is the indicator that UAV n will send task data to the remote BS e or not.

The transmission rate is calculated as

$$r^t(n,e)=B_2\log_2\left(1+\mathrm{SINR}^t(n,e)\right), \quad (13)$$

where B_2 is the transmission bandwidth of UAV n. The transmission time from UAV n to the remote BS e is thus derived as

$$T_{\mathrm{tran}}^t(n,e)=\frac{D_u^t}{r^t(n,e)}. \quad (14)$$

3.4 Computation model

The dynamic UAVs and static remote BS are used as edge servers, providing edge services for ground users. The users send task data to them for further computation. Therefore, the computations adopted for UAV and remote BS, variously, are illustrated in the forthcoming content, which elucidates the allocation of all tasks with equal computational resources.

3.4.1 Computation in UAV

UAV n selects $\mathrm{NUM}_{\mathrm{local}}^t(n)=\sum_{u=1}^U a_{u,n}^t-\sum_{u=1}^U b_{u,e}^t$ association users to be computed locally. Thus, the computation time of the selected user u is calculated as

$$T_{\mathrm{com}}^t(u,n)=\frac{D_u^t}{\frac{f_n}{\mathrm{NUM}_{\mathrm{local}}^t(n)}}, \quad (15)$$

where f_n is the computation capacity of UAV n.

3.4.2 Computation in the remote BS

All UAVs select $\mathrm{NUM}_{\mathrm{remote}}^t(e)=\sum_{n=1}^N\sum_{u=1}^U a_{u,n}^t b_{u,e}^t$ association users to be transmitted to the remote BS e. The computation time of user u in e is derived as

$$T_{\mathrm{com}}^t(u,e)=\frac{D_u^t}{\frac{f_e}{\mathrm{NUM}_{\mathrm{remote}}^t(e)}}, \quad (16)$$

where f_e is the computation capacity of the remote BS e.

3.5 Energy consumption model

The energy consumption to maintain the flight is considered in this paper. For UAV n, the energy required for flying (Chakrabarty and Langelaan, 2009; Xue, 2014) can be approximated in Eq. (17), where E_{gravity}^t, E_{kinetic}^t, and E_{drag}^t denote the energy consumed by the gravity, the kinetics, and the drag respectively, w_1, w_2, and w_3 represent the parameter

weights adopted for decreasing the consumed flight energy, m is the UAV's mass, g is the gravity coefficient, v_n^t represents the flying velocity of UAV n, ρ denotes the density of the air, C_{D} denotes the drag coefficient of the air kinetics, S is the UAV's wing area, and $\bar{R}$ is the gas constant:

$$\begin{aligned}\bar{E}^t(n) &= w_1\left(E^t_{\mathrm{gravity}}\right)+w_2\left(E^t_{\mathrm{kinetic}}\right)+w_3\left(E^t_{\mathrm{drag}}\right)\\ &= w_1\left(mgh^t_n\right)+w_2\left(\frac{1}{2}m(v^t_n)^2\right)\\ &\quad+w_3\left(\frac{1}{2}\rho(v^t_n)^2C_{\mathrm{D}}S\bar{R}\right).\end{aligned} \tag{17}$$

3.6 Problem formulation

In time slot t, the total time consumed of user u is represented as

$$\begin{aligned}T^t_{\mathrm{all}}(u) = \sum_{n=1}^{N}&\left(a^t_{u,n}T^t_{\mathrm{tran}}(u,n)+(1-b^t_{u,e})T^t_{\mathrm{com}}(u,n)\right.\\ &\left.+b^t_{u,e}\left(T^t_{\mathrm{tran}}(n,e)+T^t_{\mathrm{com}}(u,e)\right)\right).\end{aligned} \tag{18}$$

When $T^t_{\mathrm{all}}(u)\le\bar{D}^t_u$, the task of user u can be successfully finished. The energy efficiency of all users is defined as

$$\mathrm{EE}^t(\mathcal{U})=\sum_{u=1}^{U}\frac{D^t_u}{\bar{E}^t(n)}I(T^t_{\mathrm{all}}(u)\le\bar{D}^t_u), \tag{19}$$

where $I(\cdot)$ is an indicator function. When $T^t_{\mathrm{all}}(u)\le\bar{D}^t_u$ is satisfied, $I(T^t_{\mathrm{all}}(u)\le\bar{D}^t_u)=1$. Otherwise, $I(T^t_{\mathrm{all}}(u)\le\bar{D}^t_u)=0$. The optimization problem is to maximize the overall energy efficiency:

$$\begin{aligned}&\max\sum_{t=1}^{T}\mathrm{EE}^t(\mathcal{U})\\ \text{s.t. C1}:&\sum_{n=1}^{N}a^t_{u,n}\le1,\sum_{k=1}^{K}c^t_{n,k}=1,\forall u\in\mathcal{U},\\ \text{C2}:&\,b^t_{u,e}\in\{0,1\},\forall u\in\mathcal{U},\forall n\in\mathcal{N},a^t_{u,n}=1,\\ \text{C3}:&\,X_{\min}\le x^t_n\le X_{\max},Y_{\min}\le y^t_n\le Y_{\max},\\ &\,H_{\min}\le h^t_n\le H_{\max},\\ \text{C4}:&\,\|\mathrm{loc}^t_n(2\mathrm{D})-\mathrm{loc}^t_u(2\mathrm{D})\|_2\le r^t_n,\forall u\in\mathcal{U},\\ &\,a^t_{u,n}=1,\\ \text{C5}:&\,T^t_{\mathrm{all}}(u)\le\bar{D}^t_u,\forall u\in\mathcal{U}.\end{aligned} \tag{20}$$

Constraint C1 indicates that each user is associated with at most one UAV, and that all UAVs can choose only one sub-channel. Constraint C2 denotes that a user's task is computed in UAV or edge BS. Constraint C3 limits the scope for the flight of UAVs. Constraint C4 demonstrates that the associated user should be inside the radio coverage scope of a UAV. Constraint C5 means that a user's task should be fulfilled within its deadline. The optimization problem is obviously non-convex, and is thus infeasible to tackle using traditional optimization approaches. This paper will present a multi-agent collaboration scheme for energy-efficient task scheduling based on DRL.

4 A multi-agent collaboration scheme for energy efficiency optimization

Preliminary details concerning the MADDPG algorithm are provided in the supplementary materials. To address the issue of energy efficiency optimization in UAV-MEC scenarios, the flight strategy, channel selection, and task scheduling are considered for multiple UAVs' collaboration. Based on MADDPG, we propose a curiosity-driven and twin-networks-structured MADDPG (CTMADDPG) algorithm to promote the exploration and stable updating among intelligent UAVs, further improving the overall performance.

4.1 Twin networks

It is known that MADDPG is derived based on DDPG, which originates from DQN and is structured with actor–critic. Inevitably, for DQN and DDPG, Q value overestimation is a crucial problem that severely affects the performance. Q value overestimation refers to an inaccurate estimate of the Q value as being greater than the actual value. Furthermore, it leads to a poor policy update of the actor network. The root cause of overestimation consists of two aspects, namely, bootstrapping and maximizing. Bootstrapping represents an inappropriate update method of the Q network, whose aim is to estimate, by itself, the temporal difference (TD) related values, and under this method, an overestimated value incurs a higher overestimated value. Maximizing refers to the method of choosing the maximum Q value. DDPG adopts a target network to solve the issue of bootstrapping. Nevertheless, maximizing remains to be solved. To ultimately settle the issue brought by maximizing, we adopt twin critic

networks critic, critic$'$ to update the Q value (Fujimoto et al., 2018). The twin networks are equipped with the same network structure but different parameters. During the training process, both of them generate Q values. Only the smaller one can be chosen as the official Q value for calculation of the target value y. The overestimation issue brought by the maximizing method is thus settled through Q value minimization.

4.2 Curiosity mechanism

In general, most of the DRL algorithms use ϵ-greedy for strategy exploration. However, the agent does not explore the undiscovered action when it is trained after a finite number of episodes. This would lead to the consequence that the network will not converge to the optimal strategies. Furthermore, merely using ϵ-greedy is ineffective in complex collaborative tasks (Zheng et al., 2021), since these tasks require efficient exploration. To address this, we use a discounted intrinsic reward to stimulate the agent's curiosity for sufficient exploration, further facilitating an optimal convergence. The intrinsic reward encourages the agent to visit all possible environmental states and gather data as much as possible. It is designed to be negatively correlated to the count of similar observations in an episode. Hence, the agent is encouraged to choose a new action that guides it to a diverse state for effective exploration. Therefore, the reward can converge to the optimal value with motivation from the inner reward via the curiosity mechanism. A detailed illustration of our scheme is presented below.

4.3 MDP model

1. Agent. In the CTMADDPG algorithm, the UAVs serve as multiple agents. They intelligently choose actions to accommodate the dynamic environment to maximize their rewards.

2. State space. In the MDP model, the observation of each UAV in time slot t is depicted in Eq. (21) at the bottom of this page. $\{\mathrm{loc}_n^t(3\mathrm{D})\}_{n\in\mathcal{N}}$ and $\{f_n\}_{n\in\mathcal{N}}$ represent a set of 3D locations and a set of computation capacities of all UAVs, respectively. It is assumed that each UAV transmits its location and computation capacity to the other UAVs in each time slot. Thus, each UAV can observe the information of the other UAVs. $\{\mathrm{loc}_u^t(3\mathrm{D}), \mathrm{SINR}_k^t(u,n), \mathrm{task}^t(u)\}_{u\in\mathcal{U},k\in\mathcal{K},a_{u,n}^t=1,c_{n,k}^t=1}$ denotes the information of UAV n's associated users, including their 3D locations, SINR values, and task information in different time slots. There are the task data size, the number of CPU cycles, and the deadline inside $\mathrm{task}^t(u)$, as shown in Eq. (1). The state space is the union of all UAVs' observations, i.e., $s^t=\{\{o_n^t\}_{n\in\mathcal{N}}\}$.

3. Action space. The action of UAV n in time slot t is illustrated as

$$a_n^t=\{p_n^t,\theta_n^t,v_n^t,k_n^t,\{b_{u,e}^t\}_{u\in\mathcal{U},a_{u,n}^t=1}\}. \tag{22}$$

Summarily, $\{p_n^t,\theta_n^t,v_n^t\}$ denotes the set of the flying actions of UAV n, including the flying plane, flying angle, and flying velocity. Three axis planes x-y, x-h, and h-y are divided as flying planes, centering on the 3D location of UAV, i.e., $\mathrm{loc}_n^t(3\mathrm{D})$. It is noted that x-y, x-h, and h-y are constantly refreshed and generated according to the real-time 3D location in different time slots. An example of flying action is presented in the supplementary materials.

4. Reward function. The reward is an evaluation indicator to score each agent's action. Agent u is trained to maximize the total rewards. To facilitate the agent's curiosity in state exploration, the discounted intrinsic reward is added into the augmented reward (Badia et al., 2020). It is formulated as

$$r_n^t=\tilde{r}_n^t+\beta\bar{r}_n^t, \tag{23}$$

where $\tilde{r}_n^t$ represents the extrinsic reward derived from the environment, $\bar{r}_n^t$ denotes the intrinsic reward to promote the state exploration, and β is the discounted coefficient.

Extrinsic reward $\tilde{r}_n^t$ is the feedback from the environment. Since the CTMADDPG algorithm is designed to tackle the optimization problem, we design the reward to be related to the optimization goal. The reward of each agent is set to be the energy

$$o_n^t=\{\{\mathrm{loc}_n^t(3\mathrm{D})\}_{n\in\mathcal{N}},\{f_n\}_{n\in\mathcal{N}},\{\mathrm{loc}_u^t(3\mathrm{D}),\mathrm{SINR}_k^t(u,n),\mathrm{task}^t(u)\}_{u\in\mathcal{U},k\in\mathcal{K},a_{u,n}^t=1,c_{n,k}^t=1},\mathrm{loc}_e(3\mathrm{D}),f_e\}. \tag{21}$$

efficiency in time slot t, depicted as

$$\tilde{r}_n^t = \mathrm{EE}^t(\mathcal{U}) = \sum_{u=1}^{U} \frac{D_u^t}{\bar{E}^t(n)} I(T_{\mathrm{all}}^t(u) \le \bar{D}_u^t). \quad (24)$$

In the intrinsic reward mechanism, an on-line episodic buffer D' is exploited to store abstract observations with a dynamic size. Abstract observation $\bar{o}_n^{t+1}$ refers to the feature extracted by a fixed embedding network with the input of observation o_n^{t+1}, i.e., $\bar{o}_n^{t+1} = f(o_n^{t+1})$. o_n^{t+1} represents the feedback observation in time slot $t+1$ after the execution of action a_n^t in time slot t, which is used for the calculation of intrinsic reward to encourage environment state exploration.

To stimulate the agent's curiosity, intrinsic reward $\bar{r}_n^t$ is designed as

$$\bar{r}_n^t = \frac{1}{\sqrt{\mathrm{NUM}(\bar{o}_n^{t+1})}} \approx \frac{1}{\sqrt{\sum_{f_i \in N_{\bar{k}}} K(\bar{o}_n^{t+1}, f_i)} + \bar{c}}, \quad (25)$$

where $\mathrm{NUM}(\bar{o}_n^{t+1})$ is the count of visits of $\bar{o}_n^{t+1}$, approximately calculated by the sum of similarities derived from an inverse kernel of Dirac delta function $K(\bar{o}_n^{t+1}, f_i)$ with additional constant $\bar{c}$. According to Badia et al. (2020), when a Dirac delta function is used, the count approximation becomes exact but consequently does not provide generalization of exploration required for huge state space. Therefore, the inverse kernel of Dirac delta function is used in our work. The function $K(\bar{o}_n^{t+1}, f_i)$ is depicted as

$$K(\bar{o}_n^{t+1}, f_i) = \frac{\bar{\epsilon}}{\frac{d^2(\bar{o}_n^{t+1}, f_i)}{d_{\mathrm{m}}^2} + \bar{\epsilon}}, \quad (26)$$

where $\bar{\epsilon}$ is the constant for the function K, $f_i \in N_{\bar{k}}$ represents one of the $\bar{k}$ nearest neighbors, $N_{\bar{k}}$ is the set of the $\bar{k}$ nearest neighbors of $\bar{o}^{t+1}$, d_{m}^2 is the mean of the squared Euclidean distance of the $\bar{k}$ nearest neighbors, and $d^2(\bar{o}_n^{t+1}, f_i)$ represents the squared Euclidean distance between $\bar{o}_n^{t+1}$ and f_i. Details of $K(\bar{o}_n^{t+1}, f_i)$ are presented in the supplementary materials.

The summation of all UAVs' rewards is calculated as $r^t = \sum_{n=1}^{N} r_n^t$, which will be stored into the experience replay buffer D for off-line training. It is mentioned that all the performance evaluations are conducted based on the extrinsic reward rather than the augmented reward.

4.4 Pseudocode of the CTMADDPG algorithm

The pseudocode of the CTMADDPG algorithm is illustrated in Algorithm 1. For better presentation, the procedure of the algorithm is given in Fig. 2.

Algorithm 1 CTMADDPG algorithm

Initialize the experience replay buffer D
for UAV n=1 to N **do**
  Initialize the parameters of the actor network $\mu^n(*)$ and twin critic networks $Q_1^n(*), Q_2^n(*)$ with random weights ϑ^n, φ_1^n, and φ_2^n
  Initialize the parameters of target actor network $\mu'^n(*)$ and twin target critic networks $Q_1'^n(*), Q_2'^n(*)$ with weights $\vartheta'^n = \vartheta^n$, $\varphi'^n_1 = \varphi_1^n$, and $\varphi'^n_2 = \varphi_2^n$
  Initialize the fixed embedding network $\Psi^n(*)$
end for
for episode=1 to M **do**
  Initialize the state space s^t
  for UAV n=1 to N **do**
    Initialize the episodic buffer D'_n
    Store the initial abstract observation $\bar{o}_n^0$ into D'_n
  end for
  for t=1 to T **do**
    for UAV n=1 to N **do**
      if $\xi < \epsilon$ **then**
        Randomly select an action a_n^t
      else
        Add diminishing action noise ϕ to the action generated by $\mu^n(*)$, i.e., $a_n^t = \mu^n(o_n^t) + \phi$
      end if
      Execute the action a_n^t
      Obtain an extrinsic reward $\tilde{r}_n^t$ and the new observation o_n^{t+1}
      Store the abstract observation $\bar{o}_n^{t+1}$ into D'_n
      Calculate the intrinsic reward $\bar{r}_n^t$ in Eqs. (25) and (26) and derive the augmented reward r_n^t in Eq. (23)
    end for
    Store (s_t, a_t, r_t, s_{t+1}) as a transition into D
    for UAV n=1 to N **do**
      Randomly select X transitions (s_x, a_x, r_x, s_{x+1}) for off-line training
      Obtain Q values $Q_{1,n}^{\mu'}, Q_{2,n}^{\mu'}$ from the twin target critic networks and select the smaller one as the formal Q value, i.e., $Q_n^{\mu'} = \min(Q_{1,n}^{\mu'}, Q_{2,n}^{\mu'})$
      Update the parameters of the critic networks via loss function minimization in Eq. (28)
      if $\Gamma\%\iota == 0$ **then**
        Update the parameters of the actor network $\mu^n(*)$ via the policy gradient ascent method in Eq. (29)
        Update the parameters of the corresponding target networks $\mu'^n(*), Q_1'^n(*), Q_2'^n(*)$ via the soft update method in Eq. (30)
      end if
    end for
  end for
end for

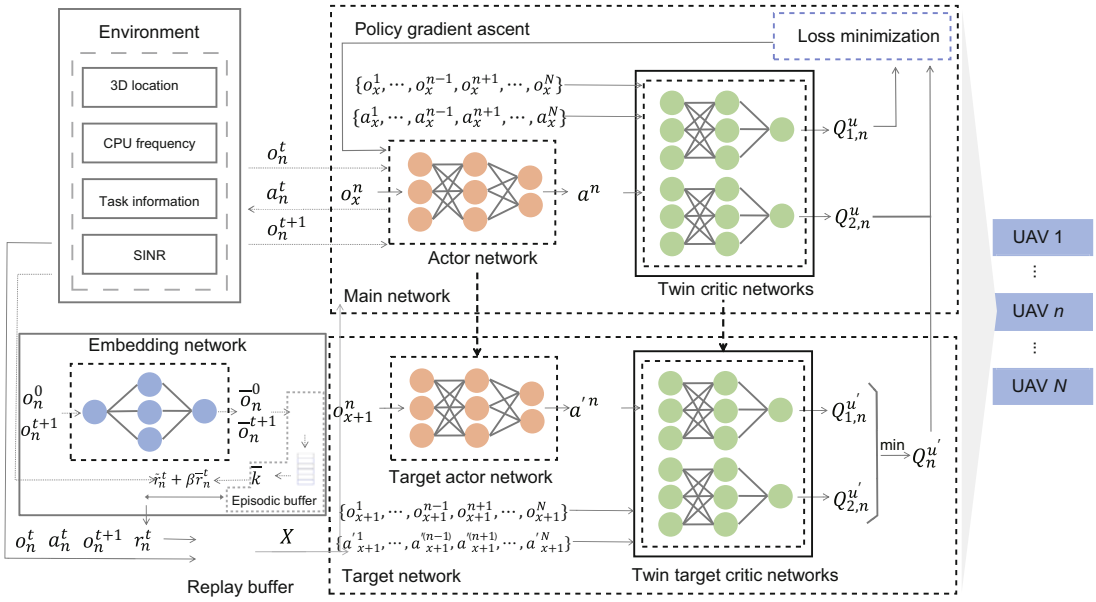

Fig. 2 Procedure of the CTMADDPG algorithm

$$y = r_x^n + \gamma Q_n^{\mu'}\left(s_{x+1}, a'^1_{x+1}, a'^2_{x+1}, \ldots, a'^N_{x+1}\right)\Big|_{a'^i_{x+1}=\mu'^i(o^i_{x+1}),\forall i\in\mathcal{N}}. \tag{27}$$

At the start of the algorithm, the initialization of experience replay buffer D and six networks $\{\mu^n(*), Q_1^n(*), Q_2^n(*), \mu'^n(*), Q_1'^n(*), Q_2'^n(*)\}$ is performed. Besides, fixed embedding network $\Psi^n(*)$ is initialized for the abstract observation and the intrinsic reward in the following training procedure (lines 1–6). M episodes are involved in the whole training process, and each of them contains T time slots (line 7). At the initial phase of each episode, state space s^t is generated from the environment. $s^t = \{o_1^t, o_2^t, ..., o_N^t\}$ consists of the initial observations of all UAVs (line 8). For each UAV, its episodic buffer D_n' is initialized to store the initial abstract observation $\bar{o}_n^0$ obtained via $\Psi^n(*)$ (lines 9–12).

In T time slots, each UAV randomly chooses an action when random probability ξ is less than ϵ. Otherwise, it adopts action a_n^t obtained from its actor network $\mu^n(*)$. To facilitate action exploration, action noise ϕ is adopted as $a_n^t = \mu^n(o_n^t) + \phi$, which diminishes with the growth of the run steps (lines 13–19). Then, each UAV executes action a_n^t and receives extrinsic reward $\tilde{r}_n^t$ and new observation o_n^{t+1} from the environment. Abstract observation $\bar{o}_n^{t+1}$ is derived from $\Psi^n(*)$ and stored into D_n'. Hence, intrinsic reward $\overline{r}_n^t$ and augmented reward r_n^t are obtained with Eqs. (23), (25), and (26) (lines 20–24). Transition (s_t, a_t, r_t, s_{t+1}) will be stored into D for off-line training (line 25). In the off-line training procedure, each UAV randomly selects X transitions to be fed into the training networks (lines 26–27). A smaller Q value $Q_n^{\mu'}$ is chosen from $Q_{1,n}^{\mu'}, Q_{2,n}^{\mu'}$, which are generated by twin target critic networks $Q'^n_1(*), Q'^n_2(*)$ (line 28). Then $Q_n^{\mu'}$ is used for calculating the target value y, as illustrated in Eq. (27) at the top of this page, where each action a'^i_{x+1} is derived from the i^{th} agent's target actor network $\mu'^i(*)$ with the input $o^i_{x+1}, \forall i \in \mathcal{N}$.

Values $Q_{1,n}^{\mu}, Q_{2,n}^{\mu}$ obtained from the twin critic networks are used for calculation of loss function. Hence, the Q values should maintain their original values to obtain accurate loss values, which are used to optimize parameters of the twin critic networks precisely. Twin critic networks $Q_1^n(*), Q_2^n(*)$ are updated via loss function minimization (line 29):

$$\begin{cases} \mathcal{L}_n^1 = \dfrac{1}{X}\displaystyle\sum_{x=1}^{X}\left(y - Q_{1,n}^{\mu}\left(s_x, a_x^1, a_x^2, \ldots, a_x^N\right)\right)^2, \\ \mathcal{L}_n^2 = \dfrac{1}{X}\displaystyle\sum_{x=1}^{X}\left(y - Q_{2,n}^{\mu}\left(s_x, a_x^1, a_x^2, \ldots, a_x^N\right)\right)^2, \end{cases} \tag{28}$$

$$\nabla_{\vartheta^n} J(\mu^n) \approx \frac{1}{X} \sum_{x=1}^{X} \Big(\nabla_{\vartheta^n} \mu^n(a^n \mid o_x^n) \nabla_{a^n} Q_{1,n}^{\mu}(s_x, a_x^1, a_x^2, \ldots, a_x^{n-1}, a^n, a_x^{n+1}, a_x^{n+2}, \ldots, a_x^N)|_{a^n=\mu^n(o_x^n)} \Big). \tag{29}$$

where $Q_{1,n}^{\mu}$ and $Q_{2,n}^{\mu}$ represent the Q values outputted by $Q_1^n(*)$ and $Q_2^n(*)$, respectively.

A delay update method is adopted to update actor network $\mu^n(*)$ every ι steps, satisfying $\Gamma \% \iota == 0$, where Γ refers to the number of training steps accumulated. This method reduces the training time and the likelihood of repeated updates. For $\mu^n(*)$, it is updated via the policy gradient ascent method (lines 30–34), shown in Eq. (29) at the top of this page, where only a^n is the output of its own actor network $\mu^n(*)$ and other actions are chosen from D. One of the twin critic networks, i.e., $Q_1^n(*)$, is selected to calculate Q value $Q_{1,n}^{\mu}$ with the input of $(s_x, a_x^1, a_x^2, \ldots, a_x^{n-1}, a^n, a_x^{n+1}, a_x^{n+2}, \ldots, a_x^N)$.

The parameters of the three mentioned target networks are refreshed via the soft update method, depicted as

$$\begin{cases} \vartheta'^n = e\vartheta^n + (1-e)\vartheta'^n, \\ \varphi_1'^n = e{\varphi_1}^n + (1-e)\varphi_1'^n, \\ \varphi_2'^n = e{\varphi_2}^n + (1-e)\varphi_2'^n, \end{cases} \tag{30}$$

where e is the proportion coefficient for soft update.

5 Performance evaluation

5.1 Parameter setting

To evaluate the performance of the proposed CTMADDPG algorithm, simulations are conducted with Python 3.6 and PyTorch 1.4 on the Ubuntu 20.4. Six benchmarks (MADDPG, MIADDPG, MASAC, MAPPO, Greedy, and Random) are used for performance evaluation. The specific settings are illustrated in the supplementary materials.

5.2 Simulation results and analysis

5.2.1 Flying trajectory of CTMADDPG with $N = 3$

To better show the 3D trajectory, the location for taking off is determined as $(1000, 0, 10)$ with the minimum height. As shown in Fig. 3a, the light and regular colors jointly represent two different trajectories of the same UAV at the convergence state. Two user clusters are distributed on the ground with

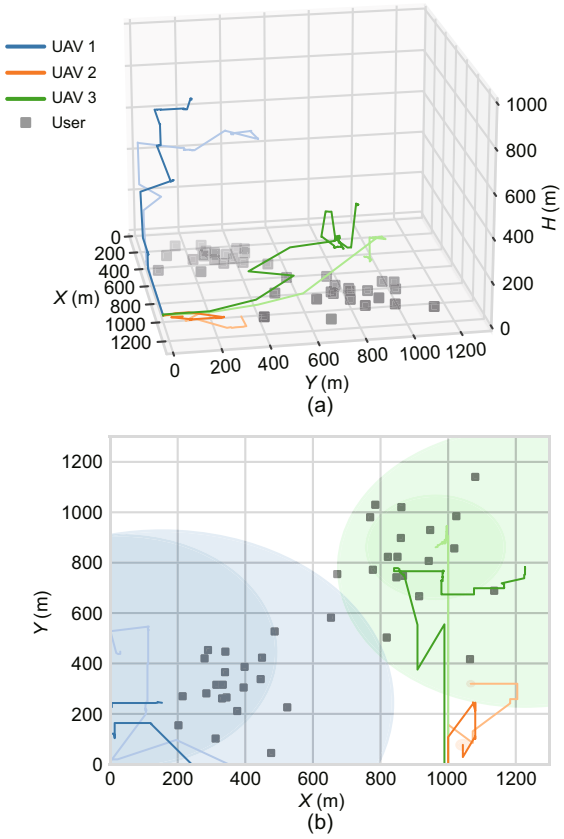

Fig. 3 Flying trajectory of CTMADDPG at the convergence state with $N = 3$: (a) flying trajectory in a 3D space; (b) flying trajectory projection in the x-y plane (References to color refer to the online version of this figure)

different task requirements in each time slot. As depicted in Fig. 3b, the projection region represents the coverage area at each UAV's final location. UAV 1 equipped with a high configuration of 20 GHz computation capacity tends to hover above the light user cluster. Contrastingly, UAV 2 and UAV 3 fly together toward the dark user cluster, both endowed with 10 GHz computation capacity. This denotes that all UAVs' flying actions are in accordance with load balance. Due to the restricted number of subchannels, i.e., $K = 2$, co-channel interference exists in the communication process. We can see that UAV 2 flies with the lowest height among the three UAVs, covering the least ground users.

5.2.2 Training rewards with $N = 3$, 4, and 5

With different numbers of UAVs, i.e., $N = 3, 4, 5$, training is conducted to study the effect of collaboration. When $N = 3, 4, 5$, we set $K = 2, 3, 4$ and $U = 40, 60, 80$, respectively. As shown in Fig. 4, the rewards of all algorithms are increasing with the increase of N. From the perspective of the CTMADDPG algorithm, its rewards in the convergence state are approximately equal to 250, 370, and 490 when N=3, 4, and 5, respectively. When there exist more users, more task data should be fulfilled. Thus, the reward is increased. Driven by curiosity, the agents in the CTMADDPG algorithm can accommodate the dynamic environment via full exploration of the environment. As for the MADDPG algorithm, the reward shows a noticeable improvement from $N = 3$ to $N = 4$. Nevertheless, it is slightly enhanced from $N = 4$ to $N = 5$. In MADDPG, more agents simultaneously participate in the training, which denotes that the environment becomes more complicated. Without enough exploration, each agent becomes stagnant and tends to prematurely converge to a sub-optimal reward. In terms of the multiple-independent-agent DDPG (MIADDPG) algorithm, its reward constantly fluctuates without convergence. The environment encountered by an agent in MIADDPG is dynamic and unknown, and accordingly it is difficult for such an agent to attain convergence.

5.2.3 Training rewards of individual UAVs of CTMADDPG with $N = 5$

Using the CTMADDPG algorithm, five UAVs simultaneously act as the learning agents to coordinately optimize the total reward. As shown in Fig. 5, the reward of each UAV is gradually growing within 1000 episodes. Eventually, they will converge to a stable value. We can see that the rewards of UAV 1, UAV 3, and UAV 5 are approximately 100. Meanwhile, the rewards of UAV 2 and UAV 4 roughly equal 60. It is inferred that not all UAVs reach the maximum value. The reason is that some UAVs need to sacrifice their interest to maximize the overall reward. If all the UAVs care only for the best reward for themselves, occupying the limited resources without

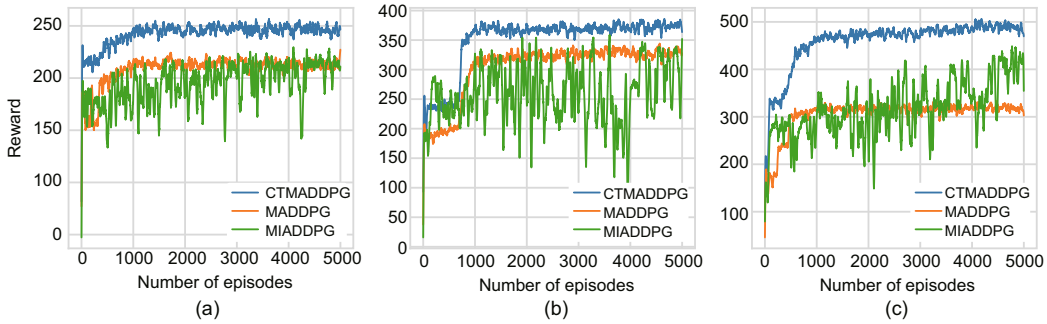

Fig. 4 Overall training reward comparison with $N = 3$ (a), $N = 4$ (b), and $N = 5$ (c)

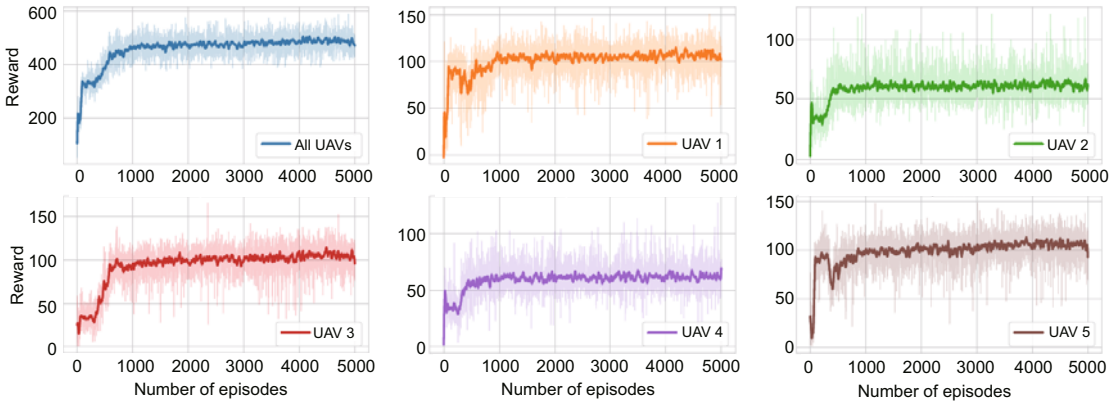

Fig. 5 Training rewards of all UAVs of CTMADDPG with $N = 5$

considering the global circumstances, none of them obtains the best results. Similar to a cooperative game, the priority of overall interest is more significant than that of individual interest. Only when the overall interest is regarded as the most important, can the UAVs obtain the most suitable result for themselves and the comprehensive system.

5.2.4 Convergence rewards with $N = 3$

Apart from the training reward, the convergence rewards with different metrics are evaluated from various aspects. The explicit analysis of the convergence rewards is provided in the supplementary materials. In summary, the CTMADDPG algorithm always obtains the highest reward compared to the six algorithms, since it can flexibly adjust its strategies with enough exploration toward the complex environment.

6 Conclusions

In this paper, we present a novel scenario in which the UAVs flexibly fly in a 3D space. They can select their flight actions related to the flying plane, flying angle, and flying velocity. Since the UAVs are deployed to provide computational services to the ground users, they need to make schedule strategies for these users. Besides, the available sub-channels are limited in this UAV-MEC scenario. There exists interference between UAVs, which has an adverse effect on communication time. When UAVs' actions are in proper alignment with the goal of maximization of the overall reward, this activates the potential for achieving an increase in the number of fulfilled tasks. To improve the overall performance, we propose a multi-agent reinforcement learning algorithm, CTMADDPG, to maximize the system energy efficiency. To facilitate the state exploration by impelling agents' curiosity to fully explore the state space, the present research ensures the involvement of the inner reward of the proposed algorithm CTMADDPG. This mechanism significantly improves the system performance. In addition, we adopt the twin critic networks to stabilize the update of the Q value. Using the proposed algorithm, all UAVs coordinate their actions to maximize the overall reward with enough state exploration. The simulation results have illustrated the outstanding performance of the CTMADDPG algorithm.

Contributors

Yang LI and Ziling WEI designed the research. Yang LI processed the data. Yang LI and Jinshu SU drafted the paper. Baokang ZHAO helped organize the paper. Yang LI, Ziling WEI, Jinshu SU, and Baokang ZHAO revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding authors upon reasonable request.

References

Al-Hourani A, Kandeepan S, Lardner S, 2014. Optimal LAP altitude for maximum coverage. *IEEE Wirel Commun Lett*, 3(6):569-572. https://doi.org/10.1109/LWC.2014.2342736

Ashraf Ateya AA, Muthanna A, Kirichek R, et al., 2019. Energy- and latency-aware hybrid offloading algorithm for UAVs. *IEEE Access*, 7:37587-37600. https://doi.org/10.1109/ACCESS.2019.2905249

Badia AP, Sprechmann P, Vitvitskyi A, et al., 2020. Never give up: learning directed exploration strategies. Proc 8th Int Conf on Learning Representations.

Chakrabarty A, Langelaan J, 2009. Energy maps for long-range path planning for small- and micro-UAVs. AIAA Guidance, Navigation, and Control Conf, Article 6113. https://doi.org/10.2514/6.2009-6113

Dai C, Zhu K, Hossain E, 2022. Multi-agent deep reinforcement learning for joint decoupled user association and trajectory design in full-duplex multi-UAV networks. *IEEE Trans Mob Comput*, 22(10):6056-6070. https://doi.org/10.1109/TMC.2022.3188473

Dai ZJ, Zhang Y, Zhang WC, et al., 2022. A multi-agent collaborative environment learning method for UAV deployment and resource allocation. *IEEE Trans Signal Inform Process Netw*, 8:120-130. https://doi.org/10.1109/TSIPN.2022.3150911

Ding CF, Wang JB, Cheng M, et al., 2023. Dynamic transmission and computation resource optimization for dense LEO satellite assisted mobile-edge computing. *IEEE Trans Commun*, 71(5):3087-3102. https://doi.org/10.1109/TCOMM.2023.3253721

Fujimoto S, van Hoof H, Meger D, 2018. Addressing function approximation error in actor-critic methods. Proc 35th Int Conf on Machine Learning, p.1587-1596.

Gu XH, Zhang GA, Wang MX, et al., 2021. UAV-aided energy-efficient edge computing networks: security offloading optimization. *IEEE Int Things J*, 9(6):4245-4258. https://doi.org/10.1109/JIOT.2021.3103391

Ji JQ, Zhu K, Yi CY, et al., 2021. Energy consumption minimization in UAV-assisted mobile-edge computing systems: joint resource allocation and trajectory design. *IEEE Int Things J*, 8(10):8570-8584. https://doi.org/10.1109/JIOT.2020.3046788

Jiang FB, Wang KZ, Dong L, et al., 2020. Deep-learning-based joint resource scheduling algorithms for hybrid MEC networks. *IEEE Int Things J*, 7(7):6252-6265. https://doi.org/10.1109/JIOT.2019.2954503

Joo S, Kang HG, Kang J, 2021. CoSMoS: cooperative sky-ground mobile edge computing system. *IEEE Trans Veh Technol*, 70(8):8373-8377. https://doi.org/10.1109/TVT.2021.3094584

Lakew DS, Tran AT, Dao NN, et al., 2023. Intelligent offloading and resource allocation in heterogeneous aerial access IoT networks. *IEEE Int Things J*, 10(7):5704-5718. https://doi.org/10.1109/JIOT.2022.3161571

Liao ZF, Ma YB, Huang JW, et al., 2021. HOTSPOT: a UAV-assisted dynamic mobility-aware offloading for mobile-edge computing in 3-D space. *IEEE Int Things J*, 8(13):10940-10952. https://doi.org/10.1109/JIOT.2021.3051214

Liu JF, Li LX, Yang FC, et al., 2019. Minimization of offloading delay for two-tier UAV with mobile edge computing. Proc 15th Int Wireless Communications & Mobile Computing Conf, p.1534-1538. https://doi.org/10.1109/IWCMC.2019.8766778

Liu Q, Shi L, Sun LL, et al., 2020. Path planning for UAV-mounted mobile edge computing with deep reinforcement learning. *IEEE Trans Veh Technol*, 69(5):5723-5728. https://doi.org/10.1109/TVT.2020.2982508

Liu XY, Xu C, Yu HB, et al., 2022. Multi-agent deep reinforcement learning for end–edge orchestrated resource allocation in industrial wireless networks. *Front Inform Technol Electron Eng*, 23(1):47-60. https://doi.org/10.1631/FITEE.2100331

Mei HB, Yang K, Liu Q, et al., 2020. Joint trajectory-resource optimization in UAV-enabled edge-cloud system with virtualized mobile clone. *IEEE Int Things J*, 7(7):5906-5921. https://doi.org/10.1109/JIOT.2019.2952677

Tun YK, Park YM, Tran NH, et al., 2021. Energy-efficient resource management in UAV-assisted mobile edge computing. *IEEE Commun Lett*, 25(1):249-253. https://doi.org/10.1109/LCOMM.2020.3026033

Wang JR, Liu KY, Pan JP, 2020. Online UAV-mounted edge server dispatching for mobile-to-mobile edge computing. *IEEE Int Things J*, 7(2):1375-1386. https://doi.org/10.1109/JIOT.2019.2954798

Wang JZ, Zhang XL, He XS, et al., 2023. Bandwidth allocation and trajectory control in UAV-assisted IoV edge computing using multiagent reinforcement learning. *IEEE Trans Reliab*, 72(2):599-608. https://doi.org/10.1109/TR.2022.3192020

Wang L, Wang KZ, Pan CH, et al., 2021. Multi-agent deep reinforcement learning-based trajectory planning for multi-UAV assisted mobile edge computing. *IEEE Trans Cogn Commun Netw*, 7(1):73-84. https://doi.org/10.1109/TCCN.2020.3027695

Wang LY, Zhang HX, Guo SS, et al., 2022. Deployment and association of multiple UAVs in UAV-assisted cellular networks with the knowledge of statistical user position. *IEEE Trans Wirel Commun*, 21(8):6553-6567. https://doi.org/10.1109/TWC.2022.3150429

Wang ZQ, Rong HG, Jiang HB, et al., 2022. A load-balanced and energy-efficient navigation scheme for UAV-mounted mobile edge computing. *IEEE Trans Netw Sci Eng*, 9(5):3659-3674. https://doi.org/10.1109/TNSE.2022.3188670

Wu SL, Xu WJ, Wang FY, et al., 2022. Distributed federated deep reinforcement learning based trajectory optimization for air-ground cooperative emergency networks. *IEEE Trans Veh Technol*, 71(8):9107-9112. https://doi.org/10.1109/TVT.2022.3175592

Xia WC, Zhu YX, De Simone L, et al., 2022. Multiagent collaborative learning for UAV enabled wireless networks. *IEEE J Sel Areas Commun*, 40(9):2630-2642. https://doi.org/10.1109/JSAC.2022.3191329

Xu S, Zhang XY, Li CG, et al., 2022. Deep reinforcement learning approach for joint trajectory design in multi-UAV IoT networks. *IEEE Trans Veh Technol*, 71(3):3389-3394. https://doi.org/10.1109/TVT.2022.3144277

Xu Y, Zhang TK, Loo J, et al., 2021. Completion time minimization for UAV-assisted mobile-edge computing systems. *IEEE Trans Veh Technol*, 70(11):12253-12259. https://doi.org/10.1109/TVT.2021.3112853

Xue NS, 2014. Design and Optimization of Lithium-Ion Batteries for Electric-Vehicle Applications. PhD Thesis, The University of Michigan, Ann Arbor, United States.

Yang L, Yao HP, Wang JJ, et al., 2020. Multi-UAV-enabled load-balance mobile-edge computing for IoT networks. *IEEE Int Things J*, 7(8):6898-6908. https://doi.org/10.1109/JIOT.2020.2971645

Yin ZY, Lin Y, Zhang YJ, et al., 2022. Collaborative multi-agent reinforcement learning aided resource allocation for UAV anti-jamming communication. *IEEE Int Things J*, 9(23):23995-24008. https://doi.org/10.1109/JIOT.2022.3188833

Yu Y, Bu XY, Yang K, et al., 2021. UAV-aided low latency multi-access edge computing. *IEEE Trans Veh Technol*, 70(5):4955-4967. https://doi.org/10.1109/TVT.2021.3072065

Yu Z, Gong YM, Gong SM, et al., 2020. Joint task offloading and resource allocation in UAV-enabled mobile edge computing. *IEEE Int Things J*, 7(4):3147-3159. https://doi.org/10.1109/JIOT.2020.2965898

Zhang L, Ansari N, 2020. Latency-aware IoT service provisioning in UAV-aided mobile-edge computing networks. *IEEE Int Things J*, 7(10):10573-10580. https://doi.org/10.1109/JIOT.2020.3005117

Zhao N, Ye ZY, Pei YY, et al., 2022. Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing. *IEEE Trans Wirel Commun*,

21(9):6949-6960.
https://doi.org/10.1109/TWC.2022.3153316

Zheng LL, Chen JR, Wang JH, et al., 2021. Episodic multi-agent reinforcement learning with curiosity-driven exploration. Proc 34th Int Conf on Neural Information Processing Systems, p.3757-3769.

Zhong RK, Liu X, Liu YW, et al., 2022. Multi-agent reinforcement learning in NOMA-aided UAV networks for cellular offloading. *IEEE Trans Wirel Commun*, 21(3):1498-1512.
https://doi.org/10.1109/TWC.2021.3104633

Zhou FH, Wu YP, Hu RQ, et al., 2018. Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems. *IEEE J Sel Areas Commun*, 36(9):1927-1941.
https://doi.org/10.1109/JSAC.2018.2864426

List of supplementary materials