Deep Reinforcement Learning Based Computation Offloading and Trajectory Planning for Multi-UAV Cooperative Target Search

Quyuan Luo, *Member, IEEE*, Tom H. Luan, *Senior Member, IEEE*, Weisong Shi, *Fellow, IEEE*, and Pingzhi Fan, *Fellow, IEEE*

***Abstract*— Unmanned aerial vehicles (UAVs) are widely used for surveillance and monitoring to complete target search tasks. However, the short battery life and moderate computational capability hinder UAVs to process computation-intensive tasks. The emerging edge computing technologies can alleviate this problem by offloading tasks to the ground edge servers. How to evaluate the search process so as to make optimal offloading decisions and make optimal flying trajectories represent fundamental research challenges. In this paper, we propose to utilize the concept of *uncertainty* to evaluate the search process, which reflects the reliability of the target search results. Thereafter, we propose a deep reinforcement learning (DRL) technique to jointly make optimal computation offloading decisions and flying orientation choices for multi-UAV cooperative target search. Specifically, we first formulate an uncertainty minimization problem based on the established system model. By introducing a reward function, we prove that the uncertainty minimization problem is equivalent to a reward maximization problem, which is further analyzed by a Markov decision process (MDP). To obtain the optimal task offloading decisions and flying orientation choices, a deep Q-network (DQN) based DRL architecture with a separated Q-network is then proposed. Finally, extensive simulations validate the effectiveness of the proposed techniques, and comprehensive discussions on how different parameters affect the search performance are given.**

***Index Terms*— Unmanned aerial vehicle, cooperative target search, edge computing, computation offloading, deep reinforcement learning (DRL).**

I. Introduction

UNMANNED aerial vehicles (UAVs) have been widely invoked for surveillance and search-related applications, to perform complicated tasks located in hazardous and even hostile environment [1]. Multi-UAVs can cooperate to perform tasks more efficiently than a single UAV [2]. Accordingly, multi-UAV cooperative target search is becoming a research hotspot and is widely used in many fields, such as agriculture, industry, disaster sensing, search & rescue, forest fire, intelligent transportation systems and crowd surveillance [3], [4], [5]. For example, the UAVs can accomplish the power line inspection in a safer and more cost-efficient manner than human patrol, allowing a fast detection of a series of defects, including line damage, cracking, galvanization loss, corrosion, and insulating breakage [6], [7]. Also, the UAVs can search mountainous areas after rainstorms to determine the potential mudslides, which could guide the timely evacuation and relocation of residents and important facilities that are seriously threatened. It is noteworthy that the term *target search* in our paper refers to not only finding targets but also achieving surveillance and anomaly detection.

However, the short battery life and moderate computational capability restrict UAVs to process computation-intensive and delay-sensitive search tasks [8]. To address this issue, the integration of edge computing into UAVs may significantly enhance the service capability of UAVs by offloading the tasks to the edge nodes [9], [10], [11], [12]. Even so, the target detection algorithms can not return absolutely credible results about the search area due to the inevitable detection errors, resulting in *uncertainty* about the target distribution in that search area. Here, *uncertainty* reflects the reliability of the target detection results. The lower the uncertainty of the search area, the higher the reliability of the target detection. By searching one area repeatedly, the uncertainty of this area can be reduced, thus improving the reliability. However, since UAVs are generally energy-limited and the search process is generally delay-sensitive, minimizing the uncertainty of search area under energy and time constrains becomes a challenge. To solve this issue, two challenges must be addressed: *1) How to optimally make offloading decisions*, so that more energy could be saved to search more areas? *2) How to dynamically plan their trajectories*, so that the uncertainty of the search area can be minimized under time constrains?

Several previous efforts have focused on the UAV-assisted edge computing framework [13], [14], [15], [16], [17]. In existing literature, UAVs are mostly considered as serving nodes with powerful communication and computation capabilities to assist the mobile devices in performing

Manuscript received 15 April 2022; revised 5 September 2022; accepted 25 October 2022. Date of publication 15 December 2022; date of current version 19 January 2023. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62101463, in part by the Natural Science Foundation of Sichuan Province of China under Grant 2022NSFSC0863, in part by the Key Research and Development Program of Sichuan under Grant 23GJHZ0209 and Grant 23ZHSF0170, and in part by the Fundamental Research Funds for the Central Universities under Grant 2682021CX044. The work of Pingzhi Fan was supported by the NSFC under Project 61731017 and Project 62020106001. *(Corresponding author: Quyuan Luo.)*

Quyuan Luo and Pingzhi Fan are with the School of Information Science and Technology, and the Provincial Key Laboratory of Information Coding and Transmission, Southwest Jiaotong University, Chengdu 611756, China (e-mail: qyluo@swjtu.edu.cn; pzfan@swjtu.edu.cn).

Tom H. Luan is with the School of Cyber Engineering, Xidian University, Xi'an 710071, China (e-mail: tom.luan@xidian.edu.cn).

Weisong Shi is with the Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716 USA (e-mail: weisong@udel.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2022.3228558.

Digital Object Identifier 10.1109/JSAC.2022.3228558

computation-intensive and latency-sensitive tasks, thus forming the UAV-assisted mobile edge computing (MEC) network. The computation offloading is mostly from the mobile users or Internet of things (IoT) users on the ground to the UAVs, and resource scheduling problems are generally formulated to minimizing either the total delay or energy consumption. Several works consider the computation offload problem where UAVs need to offload computation-intensive tasks to ground edge servers [18], [19], [20], [21], [22]. However, only perfect search of UAV is considered in existing works. For a practical search scenario, target detection algorithms can not return absolutely credible results about the search area due to inevitable detection errors, resulting in uncertainty about the target distribution in that search area.

In light of the existing works, in this article, we consider an edge computing enabled multi-UAV cooperative target search framework characterizing by imperfect search of UAVs, where UAVs themselves as users would generate much task data that should be computed locally or offloaded to the ground edge nodes. Inspired by the DRL [23], [24], we further propose a deep Q-network based task offloading decision-making and flying orientation choosing strategy, to minimize the uncertainty while ensuring energy and delay constraints. The contributions of this article are summarized as follows.

- *Model and Problem Formulation:* Considering the imperfect search of UAVs, an edge computing enabled multi-UAV cooperative target search framework is developed, where the optimal computation offloading and trajectory design are jointly considered under both energy and search time constraints. By introducing the concept of uncertainty, the cooperative target search is formulated as an uncertainty minimization problem.
- *Algorithm Design:* To obtain the optimal task offloading decision-making and flying orientation choosing actions, a Markov decision process (MDP) is first developed and a DQN based DRL architecture is then established to learn the best actions. To the best of our knowledge, this paper is the first attempt to adopt the DRL method in multi-UAV cooperative target search problem characterizing by imperfect search of UAVs.
- *Validation:* To verify the performance of our proposed algorithm, we compare our proposed algorithm with other DRL based methods including DDQN and DDPG by simulations. The superiority of our proposed algorithm is presented based on simulation results. Moreover, we also conduct extensive simulations to discuss how different parameters affect the search performance under our proposal and other DQN based schemes.

The reminder of this article is organized as follows. The related work is presented in Section II. Section III depicts the system model and uncertainty minimization problem. Section IV presents the problem analysis and MDP. In Section V, a DQN-based task offloading decision-making and flying orientation choosing strategy is proposed. In Section VI, performance evaluation results are presented. The conclusion is drawn in Section VII.

II. Related Works

In this section, we survey the existing literature on cooperative target search, computation offloading in UAV scenario, and the DRL based decision-making technique.

A. Cooperative Target Search of UAVs

Many researchers have focused on cooperative target search for UAV swarms/teams in a dynamic and risky environment. Most target search problem is formulated to find a reliable path for UAVs. Marvelous solutions have been proposed, such as grid-based search algorithm [25] and predictive algorithm [26]. With the development of intelligent systems and cooperative control theory, the target search by intelligent algorithms is emerging. By dividing the global optimization problem into several local optimization problems, an intelligent self-organized algorithm (ISOA) to solve a cooperative target search planning problem for multi-UAVs is proposed in [27]. [28] proposes a dynamic two-stage scheme by applying the concept of the closed search to multi-UAVs cooperative target search. Based on the cooperation-competition mechanism, the search algorithm was designed for the first stage. For the second stage, a search tracking approach was developed for return. Similarly, to solve the closed cooperative target search problem, [29] proposes an immune genetic algorithm to improve the search efficiency. Considering the complex constraints of multi-UAVs, [30] proposes a dynamic discrete pigeon-inspired optimization algorithm, which can reach the global optima in a discrete environment. These works ignoring the processing of computation-intensive and latency-sensitive tasks.

B. Computation Offloading and Trajectory Design for UAVs

For computation offloading, the UAVs in most existing works serve as computing nodes to assist edge systems in providing mobile users with better services [31], [32]. For example, [33] investigates the problem of task offloading from IoT mobile devices (IMDs) to the UAV, aiming to minimize the overall energy consumption for accomplishing the tasks. [34] investigates a collaborative UAV-assisted MEC systems, where UAVs are regarded as assisted edge clouds (ECs) for large-scale sparely-distributed user equipment. [35] considers that a UAV equipped with an energy transmitter (ET) and an MEC server charges sensor devices (SDs) and provides computing service for active SDs. In [36], a mobile UAV-assisted edge computing framework is considered, where a computing server mounted on a fixed-wing UAV assists the smart ground smart terminals (STs) with their computing tasks. By jointly optimizing the trajectory and CPU frequency of the UAV, the UAV's energy consumption is minimized. Similarly, [37] proposes an air-ground integrated aerial computing framework where a cloudlet server and multiple mobile edge servers are mounted on drones to provide reliable and efficient edge computing services for ground devices. To this end, the joint implementation of computation offloading and trajectory design is investigated. Considering the computation offloading, resource allocation, and flying trajectory scheduling of UAV, [38] investigates a UAV-assisted mobile edge

computing system, where a UAV equipped with an MEC server provides the computation capability for the overlaid ground smart mobile devices (SMDs).

However, UAVs also need to offload demanding computing tasks to the ground base station (GBS) when computation-intensive tasks are generated. Several efforts have been done in this area in recent years [8], [18], [19], [20], [21], [22]. From the security perspective, an energy-efficient computation offloading strategy is designed in [8] for UAV-Edge computing systems. To achieve high quality of service (QoS), [18] designs a UAV-Edge-Cloud computation offloading model for multi-UAVs, aiming to support computation-intensive tasks. In [19], UAVs are deployed in smart city for data sensing and social services. The moving vehicles are considered to assist the computation offloading for UAVs. A bargaining game is then formulated and an offloading algorithm is proposed to obtain the optimal offloading strategy. Similarly, [20] leverages the city-wide IoT infrastructure to enhance UAV's computation and communication capability. An optimal stopping time problem over a semi-Markov process is then formulated. [21] considers a problem of cooperative computation offloading for UAVs, where a UAV can offload computation workload to edge servers in MEC. [22] investigates a UAV-oriented computation offloading system, where the UAV desires to complete its onboard computation demands with the assistance of a ground edge-computing infrastructure. To this end, the UAV's longitudinal mobility, communication and computation are jointly optimized to minimize the whole energy consumption.

C. DRL-Based Decision-Making for UAVs

As to the DRL-based decision-making technique, [39] considers to choose the best communication technology (i.e., Wi-Fi or cellular) for task offloading of UAVs. To minimize the overall delay perceived by users and the energy on UAVs, a multi-agent reinforcement learning (MARL) method is proposed. In MARL, UAVs can continuously learn the best actions. In [13], the UAVs are regarded as helpers to provide value-added edge computing services. A MARL based method is proposed for the formulated computation offloading problem to determine the proper target helper and bandwidth allocation. Considering UAVs act as mobile aerial base stations in a post-disaster scenario, [40] formulates a UAV trajectory optimization problem to maximize the uplink throughput of UAV network. By transforming the original problem into a constrained Markov decision-making process, a safe-DQN-based algorithm is proposed to select the optimal actions for UAVs. Similarly, a safe-DQN-based algorithm is proposed in [41] to jointly optimize video levels selection and power allocation in a UAV-enabled video streaming scenario.

D. Summary

In summary, although marvelous solutions are proposed in existing works. Many works consider UAVs as serving nodes to provide computing services for ground mobile users or devices. In these works, joint computation offloading, resource allocation and trajectory design is mainly considered. Several works consider the computation offload problem where UAVs need to offload computation-intensive tasks to ground edge servers. The existing works mainly focus on a perfect search of UAVs. However, for a practical search scenario, target detection algorithms can not return absolutely credible results about the search area due to the inevitable detection errors, resulting in uncertainty about the target distribution in that search area. For a certain search area, UAVs may need to search it repeatedly to obtain a high-confidence results about the target distribution. Different from existing works, this paper investigate the imperfect target search scenario, where the optimal computation offloading from UAVs to ground edge servers and trajectory design are jointly considered under both energy and search time constraints. More importantly, inspired by the powerful capability of DRL in addressing complex optimization problem, a DQN-based edge computing enabled multi-UAV cooperative target search strategy is proposed. Actually, to the best of our knowledge, this paper is the first attempt to adopt the DRL method in multi-UAV cooperative target search characterizing by imperfect search of UAVs, where task offloading decisions and trajectory design are jointly optimized under energy and search time constraints.

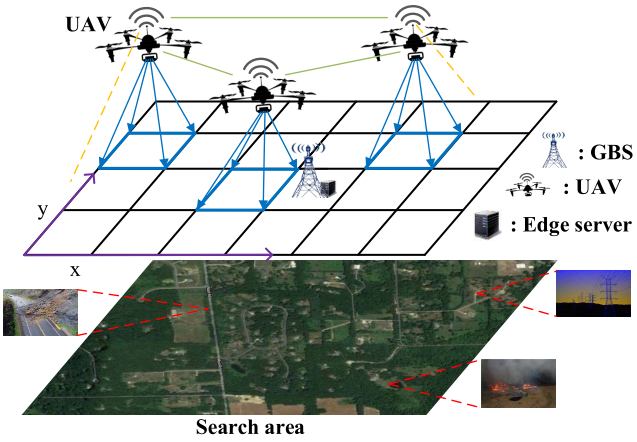


Fig. 1. Edge computing enabled multi-UAV cooperative target search framework.

III. Edge Computing Enabled Multi-UAV Cooperative Target Search Framework

A. System Description

Fig. 1 illustrates the edge computing enabled multi-UAV cooperative target search framework. We consider the search area E is a bounded $X \times Y$ area, which is further discretized and rasterized into $L_x \times L_y$ cells. The UAVs fly at a constant altitude H over these cells. A bird's view camera is mounted to the underside of each UAV and can capture the video or image of a cell. We denote the search task $\mathcal{T}_{l_x,l_y}$ in cell $[l_x, l_y]$ by two items, i.e., $\mathcal{T}_{l_x,l_y} \triangleq \{D_{l_x,l_y}, C_{l_x,l_y}\}$, where D_{l_x,l_y} denotes the data size of $\mathcal{T}_{l_x,l_y}$, and C_{l_x,l_y} denotes the processing density (in CPU cycles/bit) of $\mathcal{T}_{l_x,l_y}$. Note that the data size of $\mathcal{T}_{l_x,l_y}$ is assumed to follow a normal distribution with expectation μ such that $E(D_{l_x,l_y}) = \mu$. Assume there are N UAVs taking off from N_{off} points and must return to these points before they run out of energy. For an arbitrary

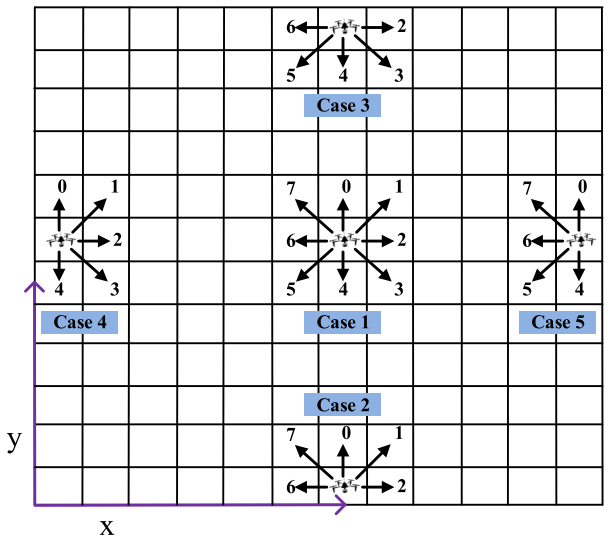


Fig. 2. Dynamics model of UAVs.

take-off point j ($1 \leq j \leq N_{off}$), its grid position is denoted by $v_j = [x_j, y_j]$, and N_j UAVs would take off from and return to j. It is noteworthy that $\sum_j N_j = N$. A ground base station (GBS) equipped with an edge server is located in the center of the search area, providing powerful computing capacities and acting as a control center for UAVs.

B. Dynamics Model of UAV

At each time step, a UAV can move from the center of its current cell to the center of one of its neighboring cells, subject to boundary constraints. The position of UAV n ($n \in \mathbb{N} = \{1, 2, \ldots, N\}$) at time step t is denoted by $v_n^t = [x_n^t, y_n^t] \in \{1, 2, \ldots, L_x\} \times \{1, 2, \ldots, L_y\}$. Correspondingly, the real position of UAV n is $[(x_n^t - 0.5)\frac{X}{L_x}, (y_n^t - 0.5)\frac{Y}{L_y}, H]$. Let $\mathcal{O}_n^t$ denote the orientation set UAV n can choose at time step t, which is defined as {0 (north), 1 (northeast), 2 (east), 3 (southeast), 4 (south), 5 (southwest), 6 (west), 7 (northwest)}. As shown in Fig. 2, there are generally five cases for different orientation choices.

- *Case 1*: When none of the UAV's eight neighboring cells has boundary constraints, the UAV can choose one orientation $o_n^t \in \{0, 1, 2, 3, 4, 5, 6, 7\}$;
- *Case 2*: When the UAV has reached the boundary of y-axis when $y_n^t = 0$, the UAV can choose one orientation $o_n^t \in \{0, 1, 2, 6, 7\}$;
- *Case 3*: When the UAV has reached the boundary of y-axis when $y_n^t = L_y$, the UAV can choose one orientation $o_n^t \in \{2, 3, 4, 5, 6\}$;
- *Case 4*: When the UAV has reached the boundary of x-axis when $x_n^t = 0$, the UAV can choose one orientation $o_n^t \in \{0, 1, 2, 3, 4\}$;
- *Case 5*: When the UAV has reached the boundary of x-axis when $x_n^t = L_x$, the UAV can choose one orientation $o_n^t \in \{0, 4, 5, 6, 7\}$;

When UAV n moves from one cell to another, the kinetic energy consumption can be modeled as

$$E_n^f(o_n^t) = \varrho V^2, \tag{1}$$

where V denotes the UAV's pre-set flying speed, ϱ is a coefficient related to flying duration $\Delta(o_n^t)$ and UAV's mass M including its payload, defined as $\varrho = 0.5M\Delta(o_n^t)$ [42]. For simplicity, we ignore UAV's weight change during flying. The flying duration $\Delta(o_n^t)$ is expressed as

$$\Delta(o_n^t) = \begin{cases} \dfrac{Y}{L_y V}, & o_n^t \in \{0, 4\}, \\ \dfrac{X}{L_x V}, & o_n^t \in \{2, 6\}, \\ \dfrac{1}{V}\sqrt{(\dfrac{X}{L_x})^2 + (\dfrac{Y}{L_y})^2}, & o_n^t \in \{1, 3, 5, 7\}. \end{cases} \tag{2}$$

It can be drawn from (2) that the flying duration when $o_n^t \in \{1, 3, 5, 7\}$ has the biggest value. To achieve time-step synchronization for all UAVs, we adjust the UAV's flying speed when $o_n^t \in \{0, 4\}$ or $o_n^t \in \{2, 6\}$ to guarantee the flying duration is the same as when $o_n^t \in \{1, 3, 5, 7\}$. In this regard, we set the length of one time step as $\Delta = \frac{1}{V}\sqrt{(\frac{X}{L_x})^2 + (\frac{Y}{L_y})^2}$. The flying speed is correspondingly adjusted to

$$V(o_n^t) = \begin{cases} \dfrac{Y}{L_y \Delta}, & o_n^t \in \{0, 4\}, \\ \dfrac{X}{L_x \Delta}, & o_n^t \in \{2, 6\}, \\ V, & o_n^t \in \{1, 3, 5, 7\}. \end{cases} \tag{3}$$

Then the kinetic energy consumption is correspondingly reformulated as

$$E_n^f(o_n^t) = 0.5M\Delta V(o_n^t)^2. \tag{4}$$

C. Search Task Computing and Offloading

We use $\alpha_n^t = 0$ to denote that the search task is processed locally by UAV n, and $\alpha_n^t = 1$ to denote that the search task is offloaded to the GBS.

1) Search Task Processed Locally: Let f_n^l denote the processing capability (i.e., the amount of CPU frequency in cycles/s [43]) at UAV i assigned for local computing. The power consumption is then modeled as

$$p_n^l = \kappa_n (f_n^l)^3, \tag{5}$$

where κ_n is a coefficient related to power in UAV n [14]. The local execution time of task $\mathcal{T}_{l_x, l_y}$ is then give by

$$T_{l_x, l_y}^{\text{local}} = \frac{\mu C_{l_x, l_y}}{f_n^l}. \tag{6}$$

Then the energy consumption of UAV i for local processing is expressed as

$$\begin{aligned} E_n^l(l_x, l_y, t) &= (1 - \alpha_n^t) p_n^l T_{l_x, l_y}^{\text{local}} \\ &= (1 - \alpha_n^t) \kappa_n \mu C_{l_x, l_y} (f_n^l)^2. \end{aligned} \tag{7}$$

2) Search Task Offloaded to GBS: Let L denote the number of orthogonal licensed channels the GBS provides, each with the bandwidth of B. We denote the GBS's grid position by $u_g = [x_g, y_g]$ and such that the real positions of the GBS is $\big[\big(x_g - 0.5\big)\frac{X}{L_x}, \big(y_g - 0.5\big)\frac{Y}{L_y}, 0\big]$. Then the distance between UAV n and the GBS at time step t is then given by

$$d_{n,g}^t = \sqrt{\Big(\big(x_n^t - x_g\big)\frac{X}{L_x}\Big)^2 + \Big(\big(y_n^t - y_g\big)\frac{Y}{L_y}\Big)^2 + H^2}. \quad (8)$$

For the air-to-ground wireless communication, the line-of-sight (LoS) channels are much more predominant [16]. We consider the wireless links between UAVs and the GBS are LoS links such that the small-scale fading and shadow are ignored, and the path loss exponent becomes 2 [15]. Accordingly, the time-varying channel power gain between UAV n and the GBS at time step t can be modeled as

$$\begin{aligned} h_{n,g}^t &= h_0 (d_{n,g}^t)^{-2} \\ &= \frac{h_0}{\Big(\big(x_n^t - x_g\big)\frac{X}{L_x}\Big)^2 + \Big(\big(y_n^t - y_g\big)\frac{Y}{L_y}\Big)^2 + H^2}, \end{aligned} \quad (9)$$

where h_0 is the channel power gain at the reference distance $d_0 = 1$ m [14]. Then the transmission rate from UAV n to the GBS is expressed as

$$R_{n,g}^t = \frac{L}{\chi^t} B \log_2\big(1 + \frac{P_n h_{n,g}^t}{\sigma^2}\big), \quad (10)$$

where P_n is the transmission power of UAV n, σ^2 is the White Gaussian noise power, χ^t is the number of UAVs choosing to transmit their tasks to the GBS, denoted by $\chi^t = \sum_{n \in \mathbb{N}} \alpha_n^t$. It is noteworthy that $R_{n,g}^t$ is time-varying as $h_{n,g}^t$ is changing during the UAV is flying from one cell to another. However, the flying distance during task transmission is much less than the flying altitude H because the transmission time is generally very short. In this regard, for simplicity, we consider $h_{n,g}^t$ keep unchanged during task transmission such that $R_{n,g}^t$ is fixed during a certain time step. Assume that all χ^t UAVs share L channels, then the energy consumption for transmitting μ bits of search task to the GBS is expressed as

$$E_n^{\mathsf{tr}}(l_x, l_y, t) = \frac{\alpha_n^t P_n \mu}{R_{n,g}^t} = \frac{\alpha_n^t \chi^t P_n \mu}{LB \log_2(1 + \frac{P_n h_{n,g}^t}{\sigma^2})}. \quad (11)$$

In general, the length and width of a cell are usually tens of meters, even more than one hundred meters. The maximum UAV's speed is tens of meters per second according to the specification of DJI's UAV product [44]. In practice, the average speed of a UAV is about a few meters per second. Accordingly the order of magnitude of the time consumption of a UAV flying from a cell to another is usually at several or even more than ten seconds. As for the computing time, it is noteworthy that the target detection algorithm usually needs very low time consumption compared with the length of one time step. Also, if the image size is high, the image can be compressed to a small size by many existing mature and efficient algorithms. However, since we do not focus on this point and would pay more attention on our proposed algorithms and strategy, for simplicity, we assume that the processing of each task takes less than one time step.

D. Objective

Since the search area E is unknown to UAVs, we consider each cell of E has an associated uncertainty $u(l_x, l_y, t) \in [0, 1]$, indicating UAV's uncertainty about the target distribution in that cell. $u(l_x, l_y, t) = 1$ means cell $[l_x, l_y]$ is a completely unknown area for UAVs at time step t. Uncertainty also reflects the reliability of the target detection results. The lower the uncertainty of the cell, the higher the reliability of the target detection. $u(l_x, l_y, t)$ will be reduced as the cell is searched repeatedly, which represents less undetected information in that cell. According to the Dempster's rule of combination and the Dempster-Shafer theory (DST) [45], once a UAV has searched a cell $[l_x, l_y]$ at time step t, the uncertainty associated with that cell is reduced at an uncertainty reduction rate λ, denoted by

$$u(l_x, l_y, t+1) = \lambda u(l_x, l_y, t), \quad (12)$$

where λ can be expressed as $\lambda = 1 - \delta$, and δ is the accuracy of the target detection.

The overall objective of our system design is to find the optimal computation offloading and trajectory planning strategy for UAVs, to minimize the uncertainty over the search area under energy and time constraints, expressed as

$$\begin{aligned} \underset{\{\alpha, o\}}{\text{minimize}} \quad & \mathcal{U} = \lim_{t \to \min\{\mathscr{T}, \mathscr{E}\}} \sum_{(l_x, l_y) \in E} u(l_x, l_y, t) \quad (13) \\ \text{s.t.} \quad & \text{C1:} \alpha_n^t \in \{0, 1\} \\ & \text{C2:} \quad o_n^t \in \mathcal{O}_n^t \\ & \text{C3:} \quad V(o_n^t) = \begin{cases} \frac{Y}{L_y \Delta}, & o_n^t \in \{0, 4\} \\ \frac{X}{L_x \Delta}, & o_n^t \in \{2, 6\} \\ V, & o_n^t \in \{1, 3, 5, 7\} \end{cases} \\ & \text{C4:} \quad \Delta * \mathscr{T} \le \Theta < \Delta * (\mathscr{T} + 1), \mathscr{T} \in N^* \\ & \text{C5:} \quad \mathscr{E} = \max_{n \in \mathbb{N}} \Big\{ \mathscr{E}_n | \sum_{t=0}^{\mathscr{E}_n} E_n^l(t) + E_n^{\mathsf{tr}}(t) \le \Phi_n^{\mathsf{t}=1} - \\ & \qquad \Phi_n^{\mathsf{ret}, \mathscr{E}_n} < \sum_{t=0}^{\mathscr{E}_n + 1} E_n^l(t) + E_n^{\mathsf{tr}}(t), \mathscr{E}_n \in N^* \Big\} \end{aligned}$$

where $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_N]$ and $o = [o_1, o_2, \ldots, o_N]$. For arbitrary $\alpha_n \in \alpha$ and $o_n \in o$, they are sequential variables, denoted by $\alpha_n = [\alpha_n^1, \alpha_n^2, \ldots, \alpha_n^{\min\{\mathscr{T}, \mathscr{E}\}}]$ and $o_n = [o_n^1, o_n^2, \ldots, o_n^{\min\{\mathscr{T}, \mathscr{E}\}}]$, respectively. In (13), $t \to \min\{\mathscr{T}, \mathscr{E}\}$ indicates that the search process will be terminated once the deadline of the search process is reached or all UAVs run out of energy, where $\mathscr{T}$ and $\mathscr{E}$ denote the maximal time steps the search process can last under time constraint and energy constraint, respectively. C1 indicates that a UAV can choose either local computing or offloading actions. C2 represents that UAVs can choose different orientations under boundary constraints. C3 shows the flying speed adjustment after choosing an orientation. C4 specifies the time constraint, where Θ is the maximal area search time, $\mathscr{T}$ is the maximal time steps. C5 specifies the energy constraint, where $\Phi_n^{\mathsf{t}=0}$

denotes the initial energy of UAV n, $\Phi_n^{\mathsf{ret},\mathscr{E}_n}$ denotes the return energy of UAV n from cell $[x_n(\mathscr{E}_n), y_n(\mathscr{E}_n)]$ at time step $\mathscr{E}_n$ to it's take-off point j_n, expressed as

$$\Phi_n^{\mathsf{ret},\mathscr{E}_n} = 0.5MVd_{j_n}|_{[x_n(\mathscr{E}_n),y_n(\mathscr{E}_n)]}, \tag{14}$$

where $d_{j_n}|_{[x_n(\mathscr{E}_n),y_n(\mathscr{E}_n)]}$ is the distance from cell $[x_n(\mathscr{E}_n), y_n(\mathscr{E}_n)]$ to take-off point j_n.

In this paper, we just consider reserving the minimum kinetic energy required for the return without considering target detection being performed by the UAV on its way back to the take-off point. This is because if target detection is performed by the UAV on its way back to the take-off point, more energy would be consumed thus that the UAV can not return to the take-off point. To address this issue, more energy should be reserved when the UAV makes return decision. However, the additional energy consumption for task transmission or local computing can not be determined when the UAV makes return decision, resulting in that the required return energy is uncertain. The designed DQN-based task offloading decision-making and flying orientation choosing strategy at each cell is based on the deterministic return energy at that cell. In reverse, the return energy which contains the energy for task transmission or local computing is also based on the proposed strategy. As a result, the task offloading decision-making, the flying orientation choosing, and the return energy determination are coupled. The uncertain return energy consumption when considering target detection being performed by the UAV on its way back to the take-off point would make the original problem more complex, which is hard to solve. Accordingly, in this paper, we just consider the minimum kinetic energy required for return and focus on the task offloading decision-making and the flying orientation choosing problems.

IV. Problem Analysis and MDP

In problem (13), the uncertainty U mainly depends on the system states and a sequential actions taken by UAVs, including task offloading decisions and orientation choices. However, the actions taken by UAVs are couple and mutually affected. For instance, if too many UAVs offload their tasks through the shared wireless channels simultaneously, longer transmitting time may occur, resulting in more energy consumption. This hinders a UAV to search more cells to reduce the uncertainty. Consequently, the task offloading decision-making issue is crucial for uncertainty reduction.

Also, among all possible orientations, a UAV should fly to a neighboring cell that can reduce the uncertainty the most. However, it may occur that several UAVs choose to fly to the same cell that can respectively reduce the uncertainties of their own uncertainty maps the most, but the uncertainty of the search area is not really reduced the most. Moreover, flying to different cells may lead to different offloading energy consumption and return energy consumption, further resulting in different uncertainty reduction. Accordingly, the flying orientation choosing issue is also crucial for our objective. The optimal task offloading decision-making and flying orientation choosing strategy forms a complex relationship, making optimization problem (13) difficult to solve. In the following, we adopt the Markov decision process (MDP) to analyze the problem.

A. State Space

We denote the system state at time step t as

$$\mathcal{S}^t = \left\{\mathcal{S}_p^t, \mathcal{S}_e^t, \mathcal{S}_d^t, \mathcal{S}_u^t\right\}, \tag{15}$$

where $\mathcal{S}_p^t$ is the position state of UAVs, $\mathcal{S}_e^t$ is the residual energy state of UAVs, $\mathcal{S}_d^t$ is the residual search time, $\mathcal{S}_u^t$ is the uncertainty state of all cells. Since the position of GBS is fixed and keeps unchanged during target search process, it will not influence the output of the DQN. Accordingly, the position state of GBS is not included in the state space. Note that the system state $\mathcal{S}^t$ is observed at the beginning of time step t. Detailed system state is defined as follows.

- $\mathcal{S}_p^t$: We define $\mathcal{S}_p^t \triangleq \left\{s_{1,p}^t, s_{2,p}^t, \ldots, s_{N,p}^t\right\}$, where $s_{n,p}^t$ ($n \in \mathbb{N}$) is the position of UAV n at time step t, expressed as $[x_n^t, y_n^t]$;
- $\mathcal{S}_e^t$: We define $\mathcal{S}_e^t \triangleq \left\{s_{1,e}^t, s_{2,e}^t, \ldots, s_{N,e}^t\right\}$, where $s_{n,e}^t$ ($n \in \mathbb{N}$) is the residual energy of UAV n at the beginning of time step t;
- $\mathcal{S}_d^t$: For the residual search time, $\mathcal{S}_d^t$ will be reduced by Δ in one time step;
- $\mathcal{S}_u^t$: We define $\mathcal{S}_u^t \triangleq \left\{s_{1,u}^t, s_{2,u}^t, \ldots, s_{L_xL_y,u}^t\right\}$, where $s_{L_x(l_y-1)+l_x,u}$ is the uncertainty of cell $[l_x, l_y]$.

B. Action Space and State Transition

We define the actions taken by UAVs at time step t as $\mathcal{A}^t \triangleq \left\{\mathcal{A}_1^t, \mathcal{A}_2^t, \ldots, \mathcal{A}_N^t\right\}$. For an arbitrary UAV n, we define as $\mathcal{A}_n^t \triangleq \left\{\alpha_n^t, o_n^t\right\}$, where $\alpha_n^t \in \{0,1\}$ and $o_n^t \in \mathcal{O}_n^t$. It is noteworthy that $\mathcal{A}^t$ is consisted of all possible α_n^t and o_n^t. The system state is transitioned as follows.

1) Position State Transition: UAV n's position $[x_n^t, y_n^t]$ at next time step $t+1$ after action o_n^t is taken can be expressed as

$$[x_n^{t+1}, y_n^{t+1}] = \begin{cases} [x_n^t, y_n^t + 1], & o_n^t = 0, y_n^t < L_y \\ [x_n^t + 1, y_n^t + 1], & o_n^t = 1, \\ & x_n^t < L_x, y_n^t < L_y \\ [x_n^t + 1, y_n^t], & o_n^t = 2, x_n^t < L_x \\ [x_n^t + 1, y_n^t - 1], & o_n^t = 3, \\ & x_n^t < L_x, y_n^t > 1 \\ [x_n^t, y_n^t - 1], & o_n^t = 4, y_n^t > 1 \\ [x_n^t - 1, y_n^t - 1], & o_n^t = 5, \\ & x_n^t > 1, y_n^t > 1 \\ [x_n^t - 1, y_n^t], & o_n^t = 6, x_n^t > 1 \\ [x_n^t - 1, y_n^t + 1], & o_n^t = 7, \\ & x_n^t > 1, y_n < L_y \end{cases} \tag{16}$$

2) Residual Energy State Transition: UAV n's residual energy at next time step $t+1$ after actions o_n^t and α_n^t are taken can be expressed as

$$\Phi_n^{\mathsf{t+1}} = \Phi_n^{\mathsf{t}} - E_n^l(t) - E_n^{\mathsf{tr}}(t) - E_n^f(o_n^t). \tag{17}$$

Note that $\Phi_n^{\mathrm{t}=0} = \Phi_n^{\mathsf{init}}$ denotes the initial energy of UAV n. To better reflect how the residual energy of UAV n changes with different actions, we formulate (17) when $t > 0$ as

$$\Phi_n^{\mathrm{t}+1} = \begin{cases} \Phi_n^{\mathrm{t}} - E_n^l - 0.5M\Delta(\frac{Y}{L_y\Delta})^2, & \alpha_n^t = 0, \\ & o_n^t \in \{0,4\} \\ \Phi_n^{\mathrm{t}} - E_n^l - 0.5M\Delta(\frac{X}{L_x\Delta})^2, & \alpha_n^t = 0, \\ & o_n^t \in \{2,6\} \\ \Phi_n^{\mathrm{t}} - E_n^l - 0.5M\Delta V^2, & \alpha_n^t = 0, \\ & o_n^t \in \{1,3,5,7\} \\ \Phi_n^{\mathrm{t}} - E_n^{tr} - 0.5M\Delta(\frac{Y}{L_y\Delta})^2, & \alpha_n^t = 1, \\ & o_n^t \in \{0,4\} \\ \Phi_n^{\mathrm{t}} - E_n^{tr} - 0.5M\Delta(\frac{X}{L_x\Delta})^2, & \alpha_n^t = 1, \\ & o_n^t \in \{2,6\} \\ \Phi_n^{\mathrm{t}} - E_n^{tr} - 0.5M\Delta V^2, & \alpha_n^t = 1, \\ & o_n^t \in \{1,3,5,7\} \end{cases} \tag{18}$$

3) Residual Search Time Transition: Let Γ^t denote the residual search time at time step t, indicating the search task will be expired Γ^t later. Then the residual search time at time step $t+1$ is expressed as

$$\Gamma^{t+1} = \Gamma^t - \Delta. \tag{19}$$

Note that $\Gamma^{t=0} = \Theta$ denotes the maximal area search time.

4) Uncertainty State Transition: According to formulas (12), the uncertainty of a cell will be reduced as the cell is searched repeatedly, leading to less undetected information in that cell. The uncertainty of cell $[l_x, l_y]$ at next time step $t+1$ can be expressed as

$$u(l_x, l_y, t+1) = \begin{cases} \lambda u(l_x, l_y, t), & \exists n \in \mathbb{N}, [x_n^{t+1}, y_n^{t+1}] = [l_x, l_y] \\ u(l_x, l_y, t), & otherwise \end{cases} \tag{20}$$

Note that if multiple UAVs[1] fly to a same cell at the same time step, the uncertainty of cell $[l_x, l_y]$ would be updated $N_{[l_x,l_y]}^{t+1}$ times, where $N_{[l_x,l_y]}^{t+1}$ denotes the number of UAVs that would fly to cell $[l_x, l_y]$ at time step t. $N_{[l_x,l_y]}^{t+1}$ can be obtained by

$$N_{[l_x,l_y]}^{t+1} = \sum_{n=1}^{N} \mathbf{1}\{x_n^{t+1} = l_x, y_n^{t+1} = l_y\}, \tag{21}$$

where $\mathbf{1}\{\tau\}$ is an indicator function which equals 1 if τ is true and 0 otherwise. The uncertainty updating in (20) can be re-written as

$$u(l_x, l_y, t+1) = \lambda^{N_{[l_x,l_y]}^{t+1}} u(l_x, l_y, t), \tag{22}$$

such that the average uncertainty over the search area at the next time step is expressed as

$$U^{t+1} = \frac{\sum_{(l_x,l_y)\in E} u(l_x, l_y, t+1)}{L_x \times L_y}, \tag{23}$$

where $L_x \times L_y$ is the number of cells of the search area.

[1]In this paper, we consider that UAVs would be flying at slightly differing altitudes in order to minimize the risk of collisions if multiple UAVs fly to a cell at the same time step.

C. Rewards

The reward in the MDP refers to the system gain of taking actions $\mathcal{A}^t$ in system state $\mathcal{S}^t$ at time step t. However, the overall objective of our system design is to minimize the uncertainty over the search area under energy and time constraints as formulated in (13). In this regard, we need to transform the overall objective to an equivalent representation containing the system gain at each time step. In this paper, we introduce a reward function to describe the uncertainty reduction when actions $\mathcal{A}^t$ are taken in system state $\mathcal{S}^t$ at time step t, defined as

$$\Upsilon^t = U^t - U^{t+1}. \tag{24}$$

The first item denotes the average uncertainty over the search area at time step t while the second item denotes the average uncertainty over the search area at time step $t+1$. The difference between the two items denotes the uncertainty reduction from state $\mathcal{S}^t$ to $\mathcal{S}^{t+1}$ when actions $\mathcal{A}^t$ are taken. To make the reward function easier to understand, we give the following case as an example. Assume the average uncertainty is 0.9 at time step t and 0.7 at time step $t+1$, then the reward equals $0.9-0.7=0.2$, which means the system gain by taking action $\mathcal{A}^t$ in system state $\mathcal{S}^t$ at time step t is 0.2. Accordingly, the optimization problem in (13) can be re-formulated as

$$\begin{aligned} \underset{\{\alpha,o\}}{\text{maximize}} \quad & \Upsilon = \sum_{t=0}^{\min\{\mathscr{T},\mathscr{E}\}-1} \Upsilon^t. \\ \text{s.t.} \quad & \mathrm{C1} \sim \mathrm{C5} \end{aligned} \tag{25}$$

Proposition 1: The optimization problem (25) is equivalent to optimization problem (13), such that the solutions for (13) are obtained once (25) is solved.

Proof: According to the definition of Υ^t in (24), we have

$$\begin{aligned} \Upsilon &= \sum_{t=0}^{\min\{\mathscr{T},\mathscr{E}\}-1} \Upsilon^t \\ &= \Upsilon^0 + \Upsilon^1 + \ldots + \Upsilon^{\min\{\mathscr{T},\mathscr{E}\}-1} \\ &= (U^0 - U^1) + (U^1 - U^2) + \ldots \\ &\quad + (U^{\min\{\mathscr{T},\mathscr{E}\}-1} - U^{\min\{\mathscr{T},\mathscr{E}\}}) \\ &= U^0 - U^{\min\{\mathscr{T},\mathscr{E}\}} \\ &= U^0 - \mathcal{U}. \end{aligned} \tag{26}$$

Since U^0 is a constant, to maximize Υ in problem (25) is equivalent to minimize $\mathcal{U}$ in problem (13). Accordingly, optimization problem (25) is equivalent to (13). *Proposition 1* is proved. ■

To promote the trajectories of UAVs to meet the boundary constraints, we introduce a penalty mechanism, where a penalty will be produced if a UAV flies out of the search area. We define a penalty function of position and orientation. For an arbitrary UAV n, the penalty is defined as

$$\mathfrak{F}_n(x_n^t, y_n^t, o_n^t) = \varepsilon \times \mathbf{1}\{\mathfrak{S}_1 \cup \mathfrak{S}_2 \cup \mathfrak{S}_3 \cup \mathfrak{S}_4\}, \tag{27}$$

where $\mathfrak{S}_1 \triangleq \{x_n^t = 0, o_n^t \in \{5,6,7\}\}$, $\mathfrak{S}_2 \triangleq \{x_n^t = L_x, o_n^t \in \{1,2,3\}\}$, $\mathfrak{S}_3 \triangleq \{y_n^t = 0, o_n^t \in \{3,4,5\}\}$, and $\mathfrak{S}_4 \triangleq \{y_n^t = L_y, o_n^t \in \{0,1,7\}\}$ denote the combination

of positions and orientations that would break the boundary constraints, ε is a penalty coefficient. Then the overall penalty at time step t is expressed as

$$\mathfrak{F}^t = \sum_{n\in\mathbb{N}} \mathfrak{F}_n(x_n^t, y_n^t, o_n^t). \tag{28}$$

Accordingly, the actual reward obtained by action $\mathcal{A}^t$ at time step t can be expressed as

$$\hat{\Upsilon}^t = \Upsilon^t - \mathfrak{F}^t. \tag{29}$$

In a MDP model, the objective is to find the optimal policy π^* that maximizes the total discounted reward over time. In other words, the immediate reward may be smaller, but the optimal policy π^* may lead to a higher profitable reward from a long-term point of view [46]. As such, the considerations we discussed above are automatically considered in an MDP model. Accordingly, the optimal strategy indicating the optimal task offloading decision-making and flying orientation choosing actions, can be defined as

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi,\mathcal{S}}\left[\sum_{t=1}^{\min\{\mathscr{T},\mathscr{E}\}-1} \hat{\Upsilon}^t\right]. \tag{30}$$

V. DQN-Based Task Offloading Decision-Making and Flying Orientation Choosing Strategy

A. From Q-Learning to Deep Q-Network

In problem (30), the state space and action space grow exponentially as the number of UAVs increases, making it hard to find the optimal strategy π^* in polynomial time [47]. It has been proved that this kind of optimization problem involving decision-making is NP-hard [48]. Moreover, the problem is further complicated by the effect of current action on the future reward.

Fortunately, the reinforcement learning is emerging as a powerful method in handling the decision-making problem [49]. The agents in reinforcement learning can learn series of actions to maximize the cumulative future reward with corresponding policies over states [50]. For a state-action pair $(\mathcal{S}^t, \mathcal{A}^t)$, the expected long-term reward following current strategy π can be expressed as a Q-value,

$$\begin{aligned} Q_\pi\left(\mathcal{S}^t, \mathcal{A}^t\right) &= \mathbb{E}\left[\sum_{i=0}^{\min\{\mathscr{T},\mathscr{E}\}-1-t} \eta^i \hat{\Upsilon}^{t+i} \middle| \left(\mathcal{S}^t, \mathcal{A}^t\right)\right] \\ &= \mathbb{E}\left[\hat{\Upsilon}^t + \eta^1 \hat{\Upsilon}^{t+1} + \ldots \middle| \left(\mathcal{S}^t, \mathcal{A}^t\right)\right] \\ &= \mathbb{E}_{\mathcal{S}^{t+1}}\left[\hat{\Upsilon}^t + \eta Q_\pi\left(\mathcal{S}^{t+1}, \mathcal{A}^{t+1}\right) \middle| \left(\mathcal{S}^t, \mathcal{A}^t\right)\right], \end{aligned} \tag{31}$$

where $0 < \eta < 1$ is a discount factor indicating the impact of future reward on current actions [51]. Then the expected maximum reward can be expressed as

$$\begin{aligned} &Q_{\pi^*}\left(\mathcal{S}^t, \mathcal{A}^t\right) \\ &\quad = \mathbb{E}_{\mathcal{S}^{t+1}}\left[\hat{\Upsilon}^t + \eta \max_{\mathcal{A}^{t+1}} Q\left(\mathcal{S}^{t+1}, \mathcal{A}^{t+1}\right) \middle| \left(\mathcal{S}^t, \mathcal{A}^t\right)\right]. \end{aligned} \tag{32}$$

To approximate the long-term reward in (32), we adopt a one-step temporal difference (TD) based Q-value updating method. Based on Bellman operator, the Q-value updating process is expressed as

$$\begin{aligned} Q\left(\mathcal{S}^t, \mathcal{A}^t\right) \leftarrow Q\left(\mathcal{S}^t, \mathcal{A}^t\right) + \varphi\Bigg[\hat{\Upsilon}^t + \eta \max_{\mathcal{A}^{t+1}} Q\left(\mathcal{S}^{t+1}, \mathcal{A}^{t+1}\right) \\ - Q\left(\mathcal{S}^t, \mathcal{A}^t\right)\Bigg], \end{aligned} \tag{33}$$

where φ denotes the learning rate.

In the Q-learning process, a table containing discrete state-action combinations and Q-values is utilized. However, the states of the multi-UAV cooperative target search may be continuous. Thus, the Q-learning approach cannot be directly implemented in solving the MDP problem (30). To compensate the limitation of Q-learning, we incorporate the deep learning with the Q-learning method, which forms the Deep Q-Network (DQN). Instead of a table, DQN uses a deep neural network as a nonlinear approximator with the ability of capturing the complex interactions among various states and actions. The inputs and outputs of the deep neural network are states and Q-values, respectively. The Q-value in (31) can be estimated as $Q(\mathcal{S}^t, \mathcal{A}^t) \approx Q(\mathcal{S}^t, \mathcal{A}^t; \theta)$, where θ denotes the parameters of the DQN. Accordingly, the optimal action in state $\mathcal{S}^t$ is the one with the maximum $Q(\mathcal{S}^t, \mathcal{A}^t; \theta)$, denoted by

$$\mathcal{A}^{t*} = \arg\max_{\mathcal{A}^t} Q(\mathcal{S}^t, \mathcal{A}^t; \theta). \tag{34}$$

B. DQN Training

To train the DQN network, a mean-squared error (MSE) based loss function is defined to minimize the difference between estimated and target Q-values, expressed as

$$\mathcal{L}(\theta^t) = \mathbb{E}\left[\left(\mathcal{Y}^t - Q\left(\mathcal{S}^t, \mathcal{A}^t; \theta^t\right)\right)^2\right], \tag{35}$$

where $\mathcal{Y}^t = \hat{\Upsilon}^t + \eta \max_{\mathcal{A}^{t+1}} Q\left(\mathcal{S}^{t+1}, \mathcal{A}^{t+1}; \theta^t\right)$ is the target Q-value and $Q(\mathcal{S}^t, \mathcal{A}^t; \theta^t)$ is the estimated Q-value, θ^t is the parameters of DQN at time step t. Differentiating the loss function with respect to the parameters, we get the gradient as

$$\nabla_{\theta^t}\mathcal{L}(\theta^t) = \mathbb{E}\left[2\left(Q\left(\mathcal{S}^t, \mathcal{A}^t; \theta^t\right) - \mathcal{Y}^t\right)\nabla_{\theta^t} Q\left(\mathcal{S}^t, \mathcal{A}^t; \theta^t\right)\right]. \tag{36}$$

Based on the gradient descent, θ^t is updated as

$$\theta^t \leftarrow \theta^t - \varpi \nabla_{\theta^t}\mathcal{L}(\theta^t), \tag{37}$$

where ϖ denotes a step size coefficient, controlling the updating step size in each iteration.

In the learning process, we adopt an experience replay technique to remove the correlations in the subsequent training samples so as to improve learning efficiency. The learned experience $e^t = (\mathcal{S}^t, \mathcal{A}^t, \hat{\Upsilon}^t, \mathcal{S}^{t+1})$ at each time step is stored in a replay memory [52]. Then, a batch of stored experience

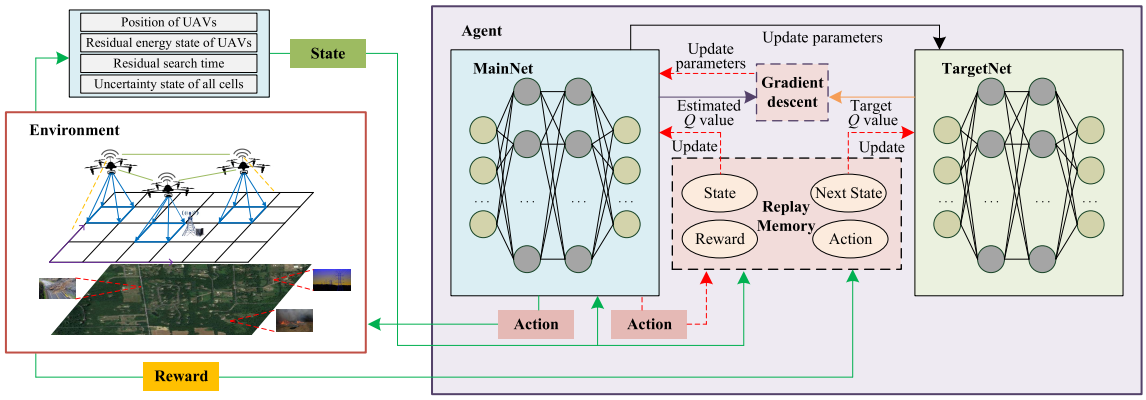


Fig. 3. DQN architecture for solving multi-UAV cooperative target search problem.

was randomly chose as samples to train the DQN. Moreover, an ϵ-greedy method is adopted before performing experience relay. With ϵ-greedy method, a random action is chosen with probability ϵ to explore better actions, otherwise, the action with maximum Q-value is chose from the replay memory.

To avoid a big correlation and oscillation between the estimated and target Q-values when the same parameters are used for Q-value calculation, a separate target Q-network is introduced for target Q-value. Let $\bar{\theta}^t$ denote the parameters of the target Q-network at time step t, the target Q-value can be correspondingly expressed as

$$\bar{\mathcal{Y}}^t = \hat{\Upsilon}^t + \eta \max_{\mathcal{A}^{t+1}} Q\left(\mathcal{S}^{t+1}, \mathcal{A}^{t+1}; \bar{\theta}^t\right). \tag{38}$$

Accordingly, $\mathcal{Y}^t$ in formulas (35) and (36) is substituted by $\bar{\mathcal{Y}}^t$. $\bar{\theta}^t$ is only updated every ζ time steps based on θ^t. Fig. 3 shows the DQN architecture for solving the task offloading decision-making and flying orientation choosing problem in the multi-UAV cooperative target search scenario, where MainNet and TargetNet refer to the neural networks used to calculate the estimated and target Q-values, respectively. Algorithm 1 shows the detailed process for training the DQN. Note that the maximum time step can be denoted by $t_{\max} = \min\{\mathscr{T}, \mathscr{E}\} - 1$ as formulated in (25).

The complexity of Algorithm 1 is mainly determined by two neural networks and one activation layer. Assuming that MainNet contained J fully connected layers and TargetNet contains K fully connected layers. The time complexity can be calculated as [53]

$$\begin{aligned} &v_{activation} n_i + \sum_{j=0}^{J-1} n_{main,j} n_{main,j+1} \\ &\quad + \sum_{k=0}^{K-1} n_{target,k} n_{target,k+1} \\ &= O\left(\sum_{j=0}^{J-1} n_{main,j} n_{main,j+1} + \sum_{k=0}^{K-1} n_{target,k} n_{target,k+1} \right), \end{aligned} \tag{39}$$

where n_i means the unit number in the i-th layer, $v_{activation}$ means the corresponding parameters determined by the type of the activation layer, $n_{main,j}$ and $n_{target,k}$ denote the unit number in the j-th DNN layer of MainNet and the k-th DNN layer of TargetNet, $n_{main,0}$ and $n_{target,0}$ equal the input size. For a fully connected layer, there is a $P \times Q$ matrix and a Q bias vector. Hence, the memory of one fully connected layer is $(P+1)Q$. The space complexity of Algorithm 1 can be accordingly calculated as [53]

$$\begin{aligned} &\sum_{j=0}^{J-1} (n_{main,j} + 1) n_{main,j+1} \\ &\quad + \sum_{k=0}^{K-1} (n_{target,k} + 1) n_{target,k+1} + \mathcal{N}(s) + N_{batch} \\ &= O\left(\sum_{j=0}^{J-1} n_{main,j} n_{main,j+1} + \sum_{k=0}^{K-1} n_{target,k} n_{target,k+1} \right) \\ &\quad + O(\mathcal{N}(s)) + O(N_{batch}), \end{aligned} \tag{40}$$

where $\mathcal{N}(s)$ is the number of the variables in the state set, N_{batch} is the mini-batch size.

VI. Performance Evaluation

A. Experimental Setup

We consider a 200 m $\times$ 200 m search area, which is further discretized and rasterized into 10×10 cells. One GBS equipped with a powerful edge server is deployed in the center of the search area. Once a search task is offloaded to the GBS, the target detection result can be obtained immediately due to the powerful computation capability of the equipped edge server. In this regard, the configuration of the edge server is ignored for simplicity. Two UAVs take off at two corners of the search area, with the cell positions of [0,0] and [9,9]. Other parameters setting of UAVs is listed in Table I. For the DQN, PyTorch [54] is used to establish the neural network. The software environment we utilize is PyTorch 1.8.0 and CUDA 11.1 with Python 3.9 on Windows 10. The GPU is NVIDIA GeForce RTX 3060 with 16GB memory. The CPU is Intel(R) Core(TM) i9-10850K with 32GB memory. Both the main and target Q-networks use a fully connected deep neural network with two hidden layers. The ReLU function is adopted as the activation function for hidden layers with 128 nodes. The learning rate and discount factor are set to

Algorithm 1 DQN-Based Task Offloading Decision-Making and Flying Orientation Choosing Strategy

Initialize replay memory
Initialize DQN with random parameters θ
Initialize target DQN with parameters $\bar{\theta} = \theta$
for episode $e = 1, \ldots, e_{\max}$ **do**
  Observe the initial state $\mathcal{S}^1$;
  for time step $t = 1, \ldots, t_{\max}$ **do**
    Choose a random probability p;
    if $p \leq \epsilon$ **then**
      Select a random action $\mathcal{A}^t$,
    else
      Choose action $\mathcal{A}^t = \arg\max_{\mathcal{A}^t} Q(\mathcal{S}^t, \mathcal{A}^t; \theta^t)$;
    end if
    Execute action $\mathcal{A}^t$, calculate $\hat{\Upsilon}^t$ and derive the next state $\mathcal{S}^{t+1}$ according to formulas (16)-(20);
    Store the experience $(\mathcal{S}^t, \mathcal{A}^t, \hat{\Upsilon}^t, \mathcal{S}^{t+1})$ in the replay memory;
    Get random mini-batch of samples $(\mathcal{S}^i, \mathcal{A}^i, \hat{\Upsilon}^i, \mathcal{S}^{i+1})$ from the replay memory;
    Calculate the target Q-value from the target DQN, $\bar{\mathcal{Y}}^i = \hat{\Upsilon}^i + \eta \max_{\mathcal{A}^{i+1}} Q(\mathcal{S}^i, \mathcal{A}^i; \bar{\theta}^t)$;
    Perform the gradient descent step on $\mathcal{L}(\theta^i) = \mathbb{E}[(\bar{\mathcal{Y}}^i - Q(\mathcal{S}^i, \mathcal{A}^i; \theta^t))^2]$ with respect to θ^t, and update θ^t;
    Every ζ time steps, update $\bar{\theta}^t$ with θ^t;
  end for
end for

TABLE I
PARAMETERS SETTING OF UAVS

Description	Value
Altitude of UAVs (H) [45]	20 m
Bandwidth per channel (B) [37]	0.1 MHz
Channel power gain (h_0) [38]	-50 dB
Length of the search area (X)	200 m
Mass of UAVs (M) [45]	0.5 $\sim$ 3 kg
Number of channels (L)	4
Number of cells along the length and width of the search area (L_x,L_y)	10,10
Number of UAVs (N)	2
Pre-set flying speed of UAVs (V) [45]	5 $\sim$ 20 m/s
Processing capability of UAVs (f_n^l) [31]	1×10^9
Processing density of tasks (C_{l_x,l_y}) [36]	2000 cycles/bit
Switched capacitance coefficient (k_n) [35]	1×10^{-24}
Transmission power of UAVs (P_n) [35]	20 dBm
Width of the search area (Y)	200 m
White Gaussian noise power (σ^2) [38]	1×10^{-9}
UAV's initial energy ($\Phi_n^{t=0}$) [45]	$\{1 \sim 6\} \times 10^4$ J
Maximal area search time (Θ) [45]	360 $\sim$ 600 s
Target detection accuracy (δ) [56]	0.2$\sim$ 0.8

0.001 and 0.95, respectively. The penalty coefficient is set to 0.05. The maximum episodes $e_{\max}$ is set to 1000. The parameter updating frequency for target Q-network is set to 100. The main parameters setting of DQN is listed in Table II.

B. Simulation Results

1) Effectiveness: We mainly consider the following four strategies:

TABLE II
PARAMETERS SETTING OF DQN

Description	Value
State vector (S^t)	$\{\mathcal{S}_p^t, \mathcal{S}_e^t, \mathcal{S}_d^t, \mathcal{S}_u^t\}$
Action vector ($\mathcal{A}^t$)	$\{\mathcal{A}_1^t, \mathcal{A}_2^t, ..., \mathcal{A}_N^t\}$
Network architecture	Fully connected neural network
Number of hidden layers	2
Number of nodes for hidden layers	128
Activation function for hidden layers	ReLU
Learning rate (φ)	0.001
Discount factor (η)	0.95
Initial ϵ	1
ϵ decay coefficient	0.995
Minimal ϵ	0.1
Mini-batch size	64
Maximum replay memory size	2000
Maximum episodes ($e_{\max}$)	3000
Updating frequency (ζ)	100
Penalty coefficient	0.05

- Our proposal, that is, the proposed task offloading decision-making and flying orientation choosing strategy aiming at minimizing the uncertainty of search area, and it is solved by DQN with separated target Q-network (recorded as *our proposal*);
- The proposed task offloading decision-making and flying orientation choosing strategy aiming at minimizing the uncertainty of search area, and it is solved by DQN without separated target Q-network (recorded as *DQN without separate target Q-network*).
- DDQN, that is, the proposed task offloading decision-making and flying orientation choosing strategy solved by Double DQN (DDQN) method [56]. DDQN has the same architecture with DQN but a more complicated target function than DQN.
- DDPG, that is, the proposed task offloading decision-making and flying orientation choosing strategy solved by Deep Deterministic Policy Gradient (DDPG) method [57]. Compared with DQN, DDPG has a actor-critic based architecture and has two additional neural networks, i.e., policy network and policy target network.

In this set of simulations, the initial energy of each UAV is set to 1×10^4 Joule, the mass and pre-set flying speed of each UAV is set to 1 kg and 5 m/s, respectively. The maximal area search time is set to 360 s, and the target detection accuracy is set to 0.4. Fig. 4 shows the changes in average uncertainty with increasing training episodes. It is obviously that the average uncertainty of our proposal, DDQN, and DDPG gradually stabilized at the optimal value in about 350 episodes of training, indicating the agent in our proposal, DDQN, and DDPG has learned the optimal task offloading decision-making and flying orientation choosing strategy to maximize the long-term reward and thus to minimize the average uncertainty. However, the scheme *DQN without separate target Q-network* requires longer episodes (approximately 600) to stabilize the average uncertainty. The average uncertainty is prone to relatively fluctuations and oscillations. This is because a big correlation between the estimated and target Q-values exists in the *DQN*

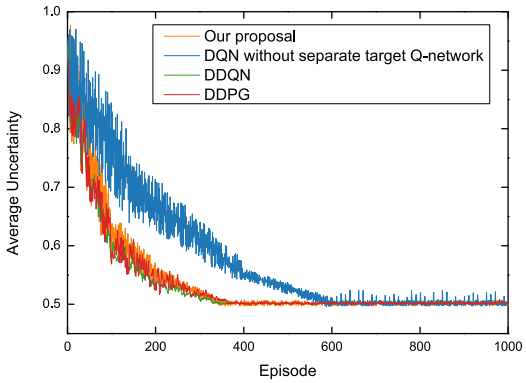


Fig. 4. Episode average uncertainty achieved during training.

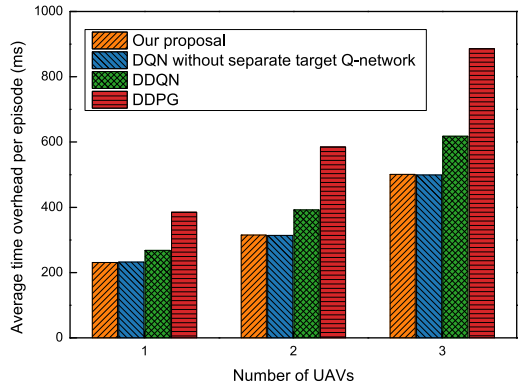


Fig. 5. Comparison of average time overhead with different schemes.

without separate target Q-network, and the estimated Q-values shift but also the target Q-values shift when the same parameters are used during the training process. The design of a separated Q-network in *our proposal* avoids these problems. Moreover, our proposal, DDQN, and DDPG have nearly the same average uncertainty performance during training. To better illustrate the performance of the four schemes. We show in Fig. 5 the comparison of average time overhead under different schemes. It can be seen that under different number of UAVs, our proposal and DQN without separate target Q-network have nearly the same average time overhead and outperform DDQN and DDPG. This is because in DDQN, the target Q-value is calculated by $\bar{\mathcal{Y}}^t = \hat{\Upsilon}^t + \eta Q(\mathcal{S}^{t+1}, \underset{a}{\text{argmax}} Q(\mathcal{S}^{t+1}, a; \theta^t), \bar{\theta}^t)$, which is more complicated than (38). For the DDPG architecture, two additional neural networks including primary network and primary target network are needed, thus resulting in more time overhead during training. Combining Fig. 4 Fig. 5, it is not hard to conclude that our proposal outperforms the benchmarks. Accordingly, in the following, we will adopt the DQN based deep reinforcement learning method to conduct more simulation to validate how other factors influence the system performance.

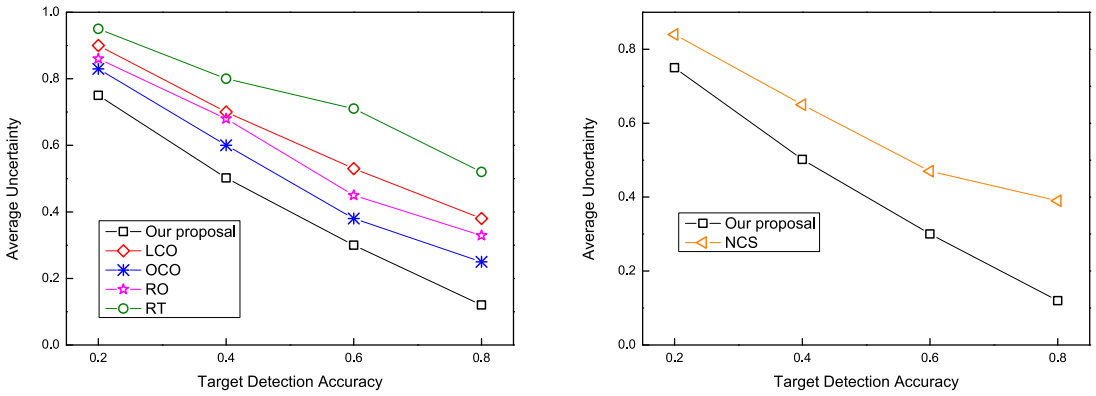
2) Average Uncertainty: We consider the following schemes as benchmarks:

- Local-Comp-Only (LCO), where all UAVs compute their computation tasks locally;
- Offload-Comp-Only (OCO), where all UAVs offload their computation tasks to GBSs;
- Random-Offload (RO), where each UAV chooses local computing or offloading randomly;
- Random-Trajectory (RT), where each UAV chooses its flying orientation randomly;
- Non-Coop-Search (NCS), where UAVs search the target area in a non-cooperative manner.

It is noteworthy that all UAVs in the LCO, OCO, RO, and RT share a common uncertainty map by communications in a cooperative manner, while each UAV in the NCS maintains a private uncertainty map without sharing it with other UAVs in a non-cooperative manner.

2.1) Effect of Target Detection Accuracy on Search Performance: According to (12), the target detection accuracy is critical to the reduction of the average uncertainty. To reveal the effect of target detection accuracy on search performance, let the target detection accuracy vary from 0.2 to 0.8, Fig. 6 is obtained. Especially, Fig. 6(a) illustrates the performance comparison between our proposal and four cooperative schemes, i.e., LCO, OCO, RO, and RT. Fig. 9(b) illustrates the performance comparison between our proposal and the non-cooperative scheme NCS. It is obviously from Fig. 9(a) that the average uncertainty of the search area decreases with the increasing target detection accuracy. Because the higher the target detection accuracy, the easier it is to identify the search area, thus leading to a lower uncertainty, which is also consistent with (12). Moreover, the average uncertainty of our proposal is lower than that of LCO, OCO, RO, and RT, which can reduce up to 68%, 52%, 64%, and 77% average uncertainty over the four schemes, respectively. This is because the UAVs in our proposal jointly make the optimal task offloading decisions and the optimal flying orientation choices learned by the agent in the DQN network for energy saving. However, more computing energy, transmission energy, or kinetic energy will be consumed in the LCO, OCO, RO, and RT schemes. Thus less area would be searched, leading to a higher uncertainty. Also, it can be seen from Fig. 9(a) that choosing offloading tasks to GBS is more beneficial to uncertainty reduction than local computing, which indicates that the offloading operation would consume less energy than local computing in most cases under this simulation scenario.

Also, the performance comparison between our proposal and NCS is shown in Fig. 6(b). In NCS, each UAV trains its DQN without the uncertainty map from other UAVs. The average uncertainty of NCS is obtained by averaging all UAVs' uncertainty map values when all UAVs finished their search tasks. It can be seen the search performance of our proposal is better than that of NCS, and can reduce up to 70% average uncertainty over NCS. This is because, during the search process, the UAVs in the NCS do not share their uncertainty maps, leading to a repeated search of a cell that has been already searched by other UAVs, ignoring these cells with a higher uncertainty. However, according to (20), a UAV should choose to fly to a cell with the highest uncertainty to reduce the uncertainty the most. Therefore, our proposal considering the uncertainty map sharing outperforms the NCS. Especially, when the target cells are searched repeatedly by UAVs, the superiority of our proposal over the non-cooperative scheme



(a) Comparison of average uncertainty with cooperative schemes (b) Comparison of average uncertainty with non-cooperative scheme

Fig. 6. Average uncertainty of search area under different accuracy of target detection.

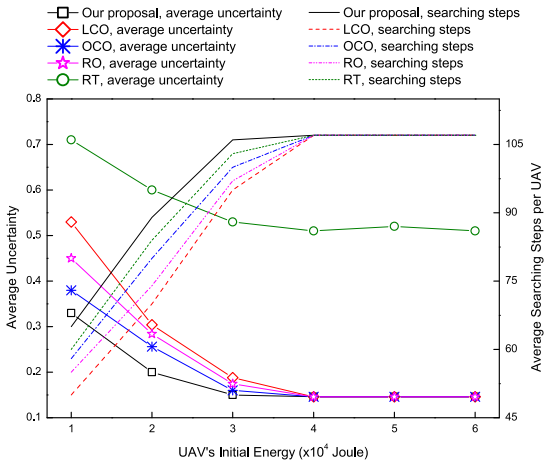
is more prominent. To verify this point, we will further show the performance gap between our proposal and NCS under different UAV's initial energy in the later simulations.

2.2) Effect of UAV's Initial Energy on Search Performance: Let the UAV's initial energy vary from 1×10^4 Joule to 6×10^4 Joule while $\delta = 0.6$ and $\Theta = 600$ s, Fig. 7 is obtained. For the comparison with cooperative schemes, Fig. 7(a) shows that the average uncertainty of all five schemes decreases with the increasing UAV's initial energy. This is because a UAV with more energy can fly to more cells to execute search tasks, leading to a reduction in uncertainty. The right Y-axis of Fig. 7 exactly shows that the average searching steps per UAV (which indicates the average number of cells searched by each UAV) increases with the increasing UAV's initial energy. Obviously, our proposal achieves the best search performance among the five cooperative schemes. It is noteworthy that the search performance of our proposal, LCO, OCO, and RO tends to be the same when UAV's initial energy gradually increases to 4×10^4 Joule. This is because when UAV's initial energy is small, energy instead of maximal area search time is the main factor limiting the search performance. Our proposal can jointly make the optimal task offloading decisions and the optimal flying orientation choices learned by the agent in the DQN network for energy saving and thus to search more cells for a better search performance. When UAV's initial energy increases, maximal area search time instead of energy is the main factor limiting a long-time area search process. The UAVs in all five schemes have nearly the same searching steps related to the maximal area search time. Considering the optimal flying orientation choosing, our proposal, LCO, OCO, and RO can obtain the same and optimal average uncertainty. However, for the RT scheme, although it has the same searching steps when UAV's initial energy gradually increases to 4×10^4 Joule, it has the worst search performance. This is because in RT, UAVs randomly choose flying orientations without considering the uncertainty minimization, leading to repeated searches on cells with low uncertainty and rare searches on cells with high uncertainty.

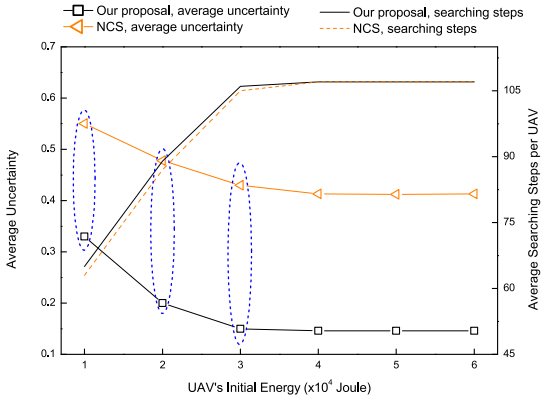
Also, Fig. 7(b) shows the comparison with the non-cooperative scheme NCS. It is obvious that the average uncertainty of both our proposal and NCS decreases with the increasing UAV's initial energy. The average uncertainty of our proposal is much better than NCS even though they have the same average searching steps. It is noteworthy that the average uncertainty performance gap between our proposal and NCS becomes bigger when UAV's initial energy increases, as circled blue in Fig. 7(b). This is because when UAV's initial energy is small, the number of target cells a UAV can search is also small, resulting in a small probability that target cells are searched repeatedly by multiple UAVs. With the increase of energy, UAVs can search for more target cells. UAVs in our proposal would fly to a cell that has the highest uncertainty to reduce the uncertainty the most through the shared uncertainty map. However, the UAVs in the NCS may search the cells that have been already searched by other UAVs repeatedly, ignoring these cells with higher uncertainties.

2.3) Effect of Maximal Area Search Time on Search Performance: UAV's initial energy has a great effect on search performance, so does the maximal area search time. To verify this point, let the maximal area search time vary from 360 s to 600 s (i.e., $6 \sim 10$ min) while $\Phi_n^{t=0} = 1 \times 10^4$ Joule, Fig. 8 is obtained. Obviously, the average uncertainty increases with the increasing maximal area search time for all six schemes. Our proposal, LCO, OCO, and RO have the same search performance. This is because the UAVs in LCO, OCO and RO can choose the optimal flying orientations as our proposal do. The task offloading decision-making has little effect on search performance when UAV's initial energy is abundant. In this case, maximal area search time is the main factor limiting the search performance. However, the average uncertainty of RT and NCS are much higher than our proposal, LCO, OCO, and RO due to the random flying orientation choosing and the respective search without uncertainty map sharing, as discussed in the previous subsection.

2.4) Joint Effect of UAV's Initial Energy and Maximal Area Search Time on Search Performance: As discussed above, both UAV's initial energy and maximal area search time have a significant effect on the duration of area search process and thus influence the search performance. How to set



(a) Comparison with cooperative schemes



(b) Comparison with non-cooperative scheme

Fig. 7. Average uncertainty under different UAV's initial energy.

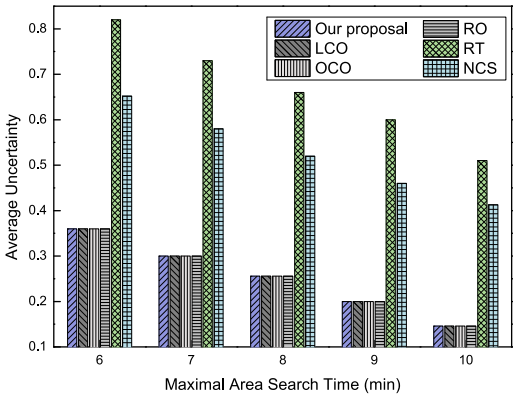


Fig. 8. Average uncertainty under different maximal area search time.

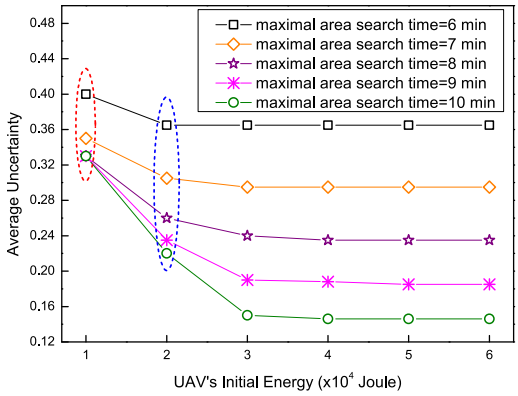


Fig. 9. Search performance under different UAV's initial energy and maximal area search time.

UAV's initial energy and maximal area search time is critical to achieve a better search performance. To explore this, let UAV's initial energy vary from 1×10^4 Joule to 6×10^4 Joule and the maximal area search time vary from 6 min to 10 min, Fig. 9 is obtained. In general, the average uncertainty decreases first and then tends to be stable for all five different area search time settings. This is because the maximal area search time would gradually becomes the main factor limiting the search performance with the increasing UAV's initial energy.

Let's look at Fig. 9 along the Y-axis. Obviously, for a certain UAV's initial energy, the average uncertainty decreases with the increasing maximal area search time. This is because a longer area search time allows that more cells with higher uncertainties can be searched by UAVs, leading to a great improvement of search performance. Especially, when UAV's initial energy is small, increasing the maximal area search time can not significantly improve the search performance. As shown in red circle when UAV's initial energy $\Phi_n^{t=0} = 1 \times 10^4$ Joule, the average uncertainty decreases first and then tends to be the same as maximal area search time increases. This indicates that UAVs end the search process ahead of the maximal area search time due to the lack of energy. More importantly, when the maximal area search time increases from 6 min to 10 min, the search performance increases more and more slowly, as the part circled blue in Fig. 9. The interesting finding indicates that once a good search performance is obtained, extending search time would not lead to a significant performance gain.

2.5) Effect of UAV's Speed on Search Performance: Let the UAV's speed vary from 5 m/s to 10 m/s, Fig. 10 is obtained. It can be seen that the average uncertainty decreases first and then increases as UAV's speed increases. Our proposal outperforms LCO, OCO, RO, RT, and NCS, which can reduce up to 28%, 17%, 26%, 38%, and 23% average uncertainty over the five schemes. It is noteworthy that too small or too high a UAV's speed would deteriorate the search performance, as shown in Fig. 10. This is because when UAV's speed is small, although the kinetic energy consumption is small according to (1), the flying duration from one cell to another is long. This results in that less cells are searched by UAVs before the maximal area search time expires, leading to a high average uncertainty. Conversely, when UAV's speed is high, the flying duration from one cell to another is short. Theoretically speaking, more cells can be searched before the maximal area search time. However, more kinetic energy would be also consumed according to (1), the remaining energy can not enable UAVs to search more cells as expected. As a result,

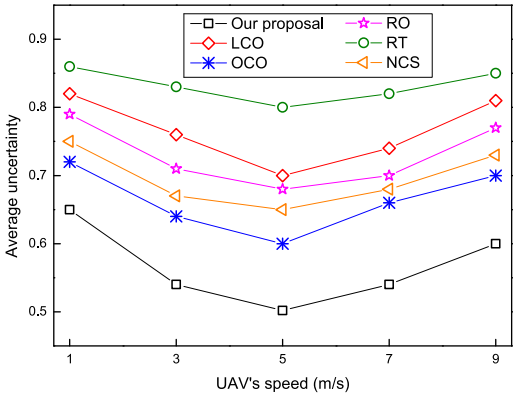


Fig. 10. Average uncertainty under different UAV's speed.

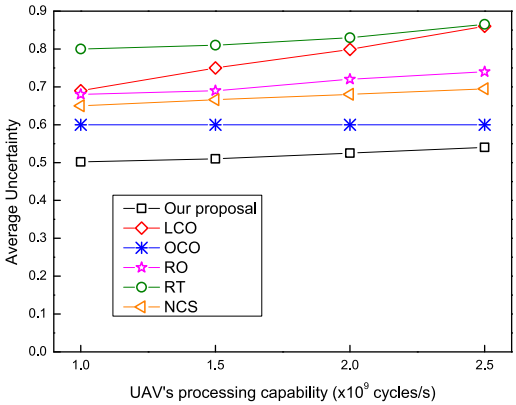


Fig. 11. Average uncertainty under different UAV's processing capability.

the search performance deteriorates. This interesting findings can guide UAVs to choose a proper flying speed to obtain a better search performance.

2.6) Effect of Processing Capability on Search Performance: We vary UAV's processing capability from 1×10^9 cycles/s to 2.5×10^9 cycles/s, Fig. 11 shows that the average uncertainty of our proposal, OCO, RO, RT, and NCS increases with the increasing UAV's processing capability while that of OCO keeps unchanged. This is because more local computing energy will be consumed when f_n^l increases based on (7). Especially, the average uncertainty of LCO is most affected by changes in UAV's processing capability. This is because local computing is the only task processing choice for LCO, and more energy will be consumed in task processing instead of flying to more cells as UAV's processing capability increases, leading to a rapid search performance deterioration.

2.7) Effect of Bandwidth on Search Performance: Let the bandwidth of each wireless communication channel vary from 0.1 MHz to 0.25 MHz, Fig. 12 shows that the average uncertainty of our proposal, OCO, RO, RT, and NCS decreases with the increasing bandwidth while that of LCO keeps unchanged. This is because there is no offloading process in LCO and the search performance has nothing to do with bandwidth. According to (11), less transmission energy would be consumed when increasing the bandwidth in our proposal, OCO, RO, RT and NCS. More energy can be used to explore more cells for uncertainty reduction. Especially, the average uncertainty of OCO is most affected by changes in bandwidth.

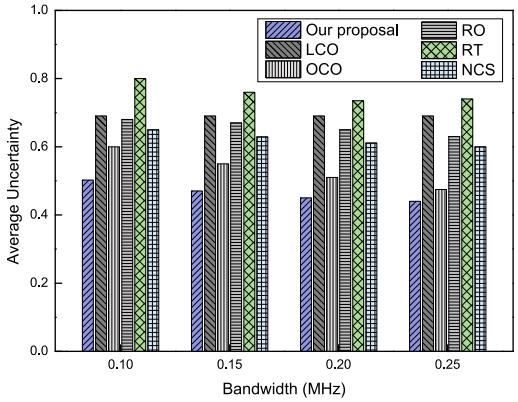
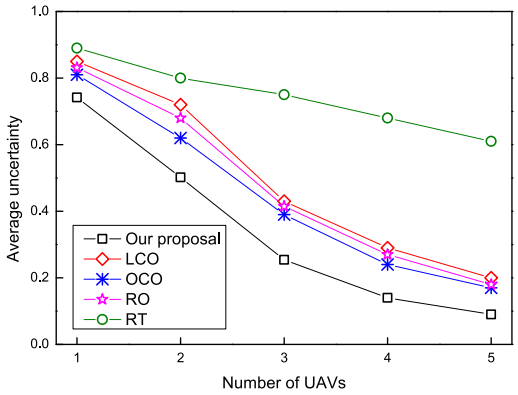


Fig. 12. Average uncertainty under different bandwidth.

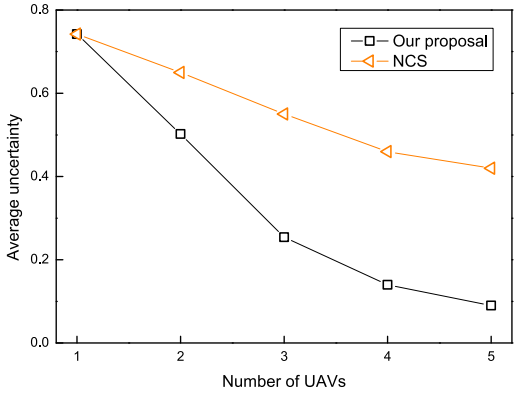
When increasing bandwidth from 0.1 MHz to 0.25 MHz, the search performance of our proposal, LCO, OCO, RO, RT, and NCS can be improved by 12%, 0%, 26%, 7%, 8%, and 8%, respectively. For all bandwidth settings, our proposal can outperforms other five schemes.

2.8) Effect of Number of RSUs on Search Performance: To show the search performance of the proposed algorithm with different number of UAVs, we vary the number of UAVs from 1 to 5. Specifically, when $N = 1$, a corner of the search area is randomly chosen as a take-off cell; when $N = 2$, two corners of the search area are randomly chosen as take-off cells; when $N = 3$, three corners of the search area are randomly chosen as take-off points; when $N = 4$, four UAVs take off at the four corners of the search area respectively; when $N = 5$, one corner of the search area is randomly chosen as the take-off cell for two UAVs. As shown in Fig. 13(a), the search performance is improved when more UAVs participate in the search process. Our proposal outperforms the cooperative benchmarks for different number of UAVs. Also, the performance comparison between our proposal and NCS is shown in Fig. 13(b). At first, the average uncertainty is the same when $N = 1$. As the number of UAVs increases, the performance gap between our proposal and NCS becomes bigger, which indicates the superiority of our proposal over non-cooperative scheme. This is This is because more UAVs in our proposal would fly to cells with higher uncertainty to reduce the uncertainty through the shared uncertainty map. More importantly, when the number of UAVs continue to increases, no big performance gain will be achieved by increasing the number of UAVs. This interesting finding indicates that once a good average uncertainty performance requirement is met, the increase of the number of UAVs would not lead to a significant performance gain but a waste of resources.

3) Search Efficiency and Offloading Proportion: To further verify the search performance of our proposal, we conduct simulations to show more details during searching process in terms of the optimal flying orientation choosing and task offloading decision-making. Here, we define search efficiency as the uniformity of searching process. As discussed above, UAVs should try to search more cells instead of just searching several cells repeatedly to minimize the uncertainty. Consequently, a good search result is the uncertainty of all cells



(a) Comparison with cooperative schemes



(b) Comparison with non-cooperative scheme

Fig. 13. Search performance under different number of UAVs.

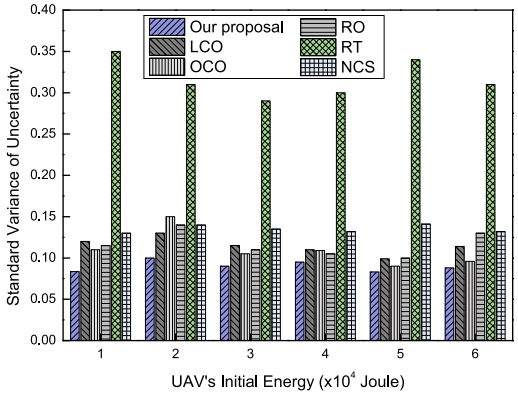


Fig. 14. Comparison of search efficiency. The standard variance of all cells' uncertainty values is used to represent the search efficiency. The lower the variance, the more efficient the search process.

should be about the same value. In this regard, we define the search efficiency as the standard variance of all cells' uncertainties. The lower the standard variance, the more efficient the search process.

Fig. 14 shows the search performance when UAV's initial energy varies from 1×10^4 Joule to 6×10^4 Joule while $\delta = 0.6$ and $\Theta = 600$ s. For all UAV's initial energy settings, our proposal has the lowest standard variance of uncertainty, indicating the best search efficiency is obtained by our proposal among six schemes. Moreover, RT has the highest standard variance of uncertainty, indicating that UAVs in RT failed to search more cells. This also further verify the effectiveness of our proposal in minimize the uncertainty by optimally choosing flying orientations. Fig. 15 shows the offloading proportion when UAV's processing capability varies from 1×10^9 cycles/s to 2.5×10^9 cycles/s. Specifically, the offloading proportions of LCO and OCO are 0% and 100%, respectively. The proportion of our proposal increases with the increasing UAV's processing capability. This is because more energy would be consumed if tasks are processed by UAVs locally when UAV's processing capability is strong. In this case, offloading is the best choice. The agent in our proposal has learned that offloading can lead to less energy consumption and thus to minimize the uncertainty. Moreover, the offloading proportion of RT and NCS is nearly the same as our proposal. This is because the optimal task offloading decision-making is also considered in RT and NCS.

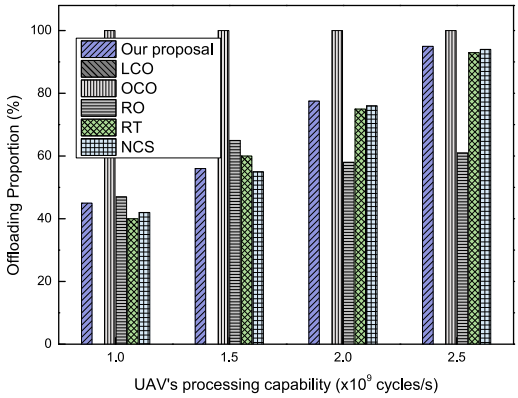


Fig. 15. Comparison of offloading proportion.

VII. Conclusion

Considering the imperfect search of UAVs, this paper investigates the edge computing enable multi-UAV cooperative target search problem. We first formulate the search process as an uncertainty minimization problem under energy and area search delay constraints. By introducing a reward function, we prove that the original problem is equivalent to a reward maximization problem. Then the interaction between different actions and rewards is analyzed by the MDP. A DRL framework is developed to efficiently optimize the task offloading decision-making and flying orientation choosing for minimizing the uncertainty over search area. Extensive simulation results validate the proposed techniques and comparison with benchmarks is also presented. Our evaluation also shows how different parameters affect the search performance, which could guide the deployment and configuration of UAVs for practical search tasks. In our future work, we will consider a more practical scenario where the movement of UAVs can be more precisely descripted by the azimuth, pitch angle and velocity in continuous space.

References

[1] L. Sun, L. Wan, and X. Wang, "Learning-based resource allocation strategy for industrial IoT in UAV-enabled MEC systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5040–5051, Sep. 2021.

[2] B. Fei, W. Bao, X. Zhu, D. Liu, T. Men, and Z. Xiao, "Autonomous cooperative search model for multi-UAV with limited communication network," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 19346–19361, Sep. 2022.

[3] J. M. D'Souza, V. V. Velpula, and K. R. Guruprasad, "Effectiveness of a camera as a UAV mounted search sensor for target detection: An experimental investigation," *Int. J. Control Autom. Syst.*, vol. 19, no. 10, pp. 1–12, 2021.

[4] D. Ebrahimi, S. Sharafeddine, P.-H. Ho, and C. Assi, "Autonomous UAV trajectory for localizing ground objects: A reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1312–1324, Apr. 2021.

[5] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource scheduling in edge computing: A survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 4, pp. 2131–2165, 4th Quart., 2021.

[6] X. Gu, D. Tang, and X. Huang, "Deep learning-based defect detection and recognition of a power grid inspection image," *Power Syst. Protection Control*, vol. 49, no. 5, pp. 91–97, 2021.

[7] Z. Zhou, C. Zhang, C. Xu, F. Xiong, Y. Zhang, and T. Umer, "Energy-efficient industrial internet of UAVs for power line inspection in smart grid," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2705–2714, Jun. 2018.

[8] T. Bai, J. Wang, Y. Ren, and L. Hanzo, "Energy-efficient computation offloading for secure UAV-edge-computing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 6074–6087, Apr. 2019.

[9] S. Garg, A. Singh, S. Batra, N. Kumar, and L. T. Yang, "UAV-empowered edge computing environment for cyber-threat detection in smart vehicles," *IEEE Netw.*, vol. 32, no. 3, pp. 42–51, May/Jun. 2018.

[10] Y. Yang, "Multi-tier computing networks for intelligent IoT," *Nature Electron.*, vol. 2, no. 1, pp. 4–5, 2019.

[11] K. Wang et al., "Task offloading with multi-tier computing resources in next generation wireless networks," 2022, *arXiv:2205.13866*.

[12] X. Liu, J. Jin, and F. Dong, "Edge-computing-based intelligent IoT: Architectures, algorithms and applications," *Sensors*, vol. 22, no. 12, p. 4464, 2022.

[13] S. Zhu, L. Gui, D. Zhao, N. Cheng, Q. Zhang, and X. Lang, "Learning-based computation offloading approaches in UAVs-assisted edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 928–944, Jan. 2021.

[14] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, Sep. 2018.

[15] H. Guo and J. Liu, "UAV-enhanced intelligent offloading for Internet of Things at the edge," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2737–2746, Apr. 2020.

[16] J. Zhang et al., "Computation-efficient offloading and trajectory scheduling for multi-UAV assisted mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2114–2125, Feb. 2019.

[17] Q. Zhang, J. Chen, L. Ji, Z. Feng, Z. Han, and Z. Chen, "Response delay optimization in mobile edge computing enabled UAV swarm," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3280–3295, Mar. 2020.

[18] W. Chen, B. Liu, H. Huang, S. Guo, and Z. Zheng, "When UAV swarm meets edge-cloud computing: The QoS perspective," *IEEE Netw.*, vol. 33, no. 2, pp. 36–43, Mar./Apr. 2019.

[19] M. Dai, Z. Su, Q. Xu, and N. Zhang, "Vehicle assisted computing offloading for unmanned aerial vehicles in smart city," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1932–1944, Mar. 2021.

[20] D. Callegaro and M. Levorato, "Optimal edge computing for infrastructure-assisted UAV systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1782–1792, Feb. 2021.

[21] S. Zhu, L. Gui, J. Chen, Q. Zhang, and N. Zhang, "Cooperative computation offloading for UAVs: A joint radio and computing resource allocation approach," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, Jul. 2018, pp. 74–79.

[22] J. Zhou, D. Tian, Z. Sheng, X. Duan, and X. Shen, "Joint mobility, communication and computation optimization for UAVs in air-ground cooperative networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 3, pp. 2493–2507, Mar. 2021.

[23] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Collaborative data scheduling for vehicular edge computing via deep reinforcement learning," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9637–9650, Oct. 2020.

[24] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proc. IEEE*, vol. 107, no. 8, pp. 1655–1674, Aug. 2019.

[25] J. Gu, T. Su, Q. Wang, X. Du, and M. Guizani, "Multiple moving targets surveillance based on a cooperative network for multi-UAV," *IEEE Commun. Mag.*, vol. 56, no. 4, pp. 82–89, Apr. 2018.

[26] J. R. Riehl, G. E. Collins, and J. P. Hespanha, "Cooperative search by UAV teams: A model predictive approach using dynamic graphs," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, no. 4, pp. 2637–2656, Oct. 2011.

[27] Z. Zhen, D. Xing, and G. Chen, "Cooperative search-attack mission planning for multi-UAV based on intelligent self-organized algorithm," *Aerosp. Sci. Technol.*, vol. 76, pp. 402–411, May 2018.

[28] D. Luo, J. Shao, Y. Xu, Y. You, and H. Duan, "Coevolution pigeon-inspired optimization with cooperation-competition mechanism for multi-UAV cooperative region search," *Appl. Sci.*, vol. 9, no. 5, p. 827, 2019.

[29] Z. Zhou, D. Luo, J. Shao, Y. Xu, and Y. You, "Immune genetic algorithm based multi-UAV cooperative target search with event-triggered mechanism," *Phys. Commun.*, vol. 41, Sep. 2020, Art. no. 101103.

[30] H. Duan, J. Zhao, Y. Deng, Y. Shi, and X. Ding, "Dynamic discrete pigeon-inspired optimization for multi-UAV cooperative search-attack mission planning," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 57, no. 1, pp. 706–720, Feb. 2021.

[31] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4576–4589, Sep. 2019.

[32] X. Huang, X. Yang, Q. Chen, and J. Zhang, "Task offloading optimization for UAV-assisted fog-enabled Internet of Things networks," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1082–1094, Sep. 2021.

[33] J. Xiong, H. Guo, and J. Liu, "Task offloading in UAV-aided edge computing: Bit allocation and trajectory optimization," *IEEE Commun. Lett.*, vol. 23, no. 3, pp. 538–541, Mar. 2019.

[34] N. Zhao, Z. Ye, Y. Pei, Y.-C. Liang, and D. Niyato, "Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 6949–6960, Sep. 2022.

[35] Y. Liu, K. Xiong, Q. Ni, P. Fan, and K. Ben Letaief, "UAV-assisted wireless powered cooperative mobile edge computing: Joint offloading, CPU control, and trajectory optimization," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2777–2790, Apr. 2020.

[36] C. Sun, W. Ni, and X. Wang, "Joint computation offloading and trajectory planning for UAV-assisted edge computing," *IEEE Trans. Wireless Commun.*, vol. 20, no. 8, pp. 5343–5358, Aug. 2021.

[37] S. Zhang, J. Liu, Y. Zhu, and J. Zhang, "Joint computation offloading and trajectory design for aerial computing," *IEEE Wireless Commun.*, vol. 28, no. 5, pp. 88–94, Nov. 2021.

[38] J. Zhang et al., "Stochastic computation offloading and trajectory scheduling for UAV-assisted mobile edge computing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3688–3699, Apr. 2019.

[39] A. Sacco, F. Esposito, G. Marchetto, and P. Montuschi, "Sustainable task offloading in UAV networks via multi-agent reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 70, no. 5, pp. 5003–5015, May 2021.

[40] T. Zhang, J. Lei, Y. Liu, C. Feng, and A. Nallanathan, "Trajectory optimization for UAV emergency communication with limited user equipment energy: A safe-DQN approach," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1236–1247, Sep. 2021.

[41] Q. Zhang, J. Miao, Z. Zhang, F. R. Yu, F. Fu, and T. Wu, "Energy-efficient video streaming in UAV-enabled wireless networks: A safe-DQN approach," in *Proc. IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–7.

[42] A. Filippone, *Flight Performance of Fixed and Rotary Wing Aircraft*. Amsterdam, The Netherlands: Elsevier, 2006.

[43] Q. Luo, C. Li, T. H. Luan, W. Shi, and W. Wu, "Self-learning based computation offloading for internet of vehicles: Model and algorithm," *IEEE Trans. Wireless Commun.*, vol. 20, no. 9, pp. 5913–5925, Sep. 2021.

[44] DJI. *Consumer Drones Comparison*. Accessed: Jan. 10, 2022. [Online]. Available: https://www.dji.com/products/compare-consumer-drones

[45] Y. Yang, M. M. Polycarpou, and A. A. Minai, "Multi-UAV cooperative search using an opportunistic learning method," *J. Dyn. Syst., Meas., Control*, vol. 129, no. 5, pp. 716–728, 2007.

[46] X. Qiu, W. Zhang, W. Chen, and Z. Zheng, "Distributed and collective deep reinforcement learning for computation offloading: A practical perspective," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 5, pp. 1085–1101, May 2021.

[47] O. A. Maillard, T. A. Mann, and S. Mannor, "How hard is my MDP? 'The distribution-norm to the rescue,"' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1835–1843.

[48] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[49] K. Wang, Y. Zhou, Z. Liu, Z. Shao, X. Luo, and Y. Yang, "Online task scheduling and resource allocation for intelligent NOMA-based industrial Internet of Things," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 803–815, May 2020.
[50] K. Zhang, Y. Zhu, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Deep learning empowered task offloading for mobile edge computing in urban informatics," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7635–7647, Oct. 2019.
[51] K. Wang, Y. Zhou, Y. Yang, X. Yuan, and X. Luo, "Task offloading in NOMA-based fog computing networks: A deep Q-learning approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.
[52] M. Volodymyr et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[53] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Deep deterministic policy gradient (DDPG)-based energy harvesting wireless communications," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8577–8588, Oct. 2019.
[54] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.
[55] A. Symington, S. Waharte, S. Julier, and N. Trigoni, "Probabilistic target detection by camera-equipped UAVs," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 4076–4081.
[56] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," 2015, *arXiv:1509.06461*.
[57] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

Quyuan Luo (Member, IEEE) received the Ph.D. degree in communication and information system from Xidian University, Xi'an, China, in 2020. He had been a Visiting Scholar in computer science with Wayne State University, USA, from 2019 to 2020. He is currently an Assistant Professor with the Provincial Key Laboratory of Information Coding and Transmission, and with the School of Information Science and Technology, Southwest Jiaotong University, China. His current research interests include intelligent transportation systems, content distribution, edge computing, and resource allocation in vehicular networks. He was a recipient of the Excellent Doctoral Dissertation Award of China Education Society of Electronics in 2021.

Tom H. Luan (Senior Member, IEEE) received the B.Eng. degree from Xi'an Jiaotong University, Xi'an, China, in 2004, the M.Phil. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2007, and the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, in 2012. He is currently a Professor with the School of Cyber Engineering, Xidian University, Xi'an. He has authored/coauthored around 60 journal articles and 30 technical papers in conference proceedings, and awarded one U.S. patent. His research interests include content distribution and media streaming in the vehicular ad hoc networks and peer-to-peer networking, protocol design, and performance evaluation of wireless cloud computing and fog computing.

Weisong Shi (Fellow, IEEE) received the B.S. degree in computer engineering from Xidian University, Xi'an, China, in 1995, and the Ph.D. degree in computer engineering from the Chinese Academy of Sciences in 2000. He is currently a Professor and the Chair of the Department of Computer and Information Sciences, University of Delaware (UD), where he leads the Connected and Autonomous Research (CAR) Laboratory. Before he joins UD, he was a Professor at Wayne State University (2002–2022). He is an internationally renowned expert in edge computing, autonomous driving and connected health. His pioneer paper entitled "Edge Computing: Vision and Challenges" has been cited more than 5000 times. He has published more than 270 papers in peer-reviewed journals and conferences. He is a Distinguished Member of ACM. He served in editorial roles for more than ten academic journals and publications, including the Editor-in-Chief of *Smart Health* and the Associate Editor-in-Chief of *IEEE Internet Computing Magazine*. More information can be found at: http://weisongshi.org.

Pingzhi Fan (Fellow, IEEE) received the M.Sc. degree in computer science from Southwest Jiaotong University, China, in 1987, and the Ph.D. degree in electronic engineering from Hull University, U.K., in 1994. He has been a Visiting Professor with Leeds University, U.K., since 1997, and a Guest Professor with Shanghai Jiaotong University since 1999. He is currently a Distinguished Professor and the Director of the Institute of Mobile Communications, Southwest Jiaotong University. He has published over 300 international journal articles and eight books (including edited), and is the inventor of 25 granted patents. His current research interests include vehicular communications, massive multiple access, and coding techniques. He also served as an EXCOM Member for IEEE Region 10, IET (IEE) Council, and IET Asia Pacific Region. He is a fellow of IET, CIE, and CIC. He was a recipient of the U.K. ORS Award in 1992, the NSFC Outstanding Young Scientist Award in 1998, the IEEE VTS Jack Neubauer Memorial Award in 2018, and the 2018 IEEE SPS Signal Processing Letters Best Paper Award. He served as the General Chair or a TPC Chair for a number of international conferences, including VTC 2016Spring, IWSDA 2019, and ITW 2018. He is the Founding Chair of the IEEE Chengdu (CD) Section, the IEEE VTS BJ Chapter, and the IEEE ComSoc CD Chapter. He is an IEEE VTS Distinguished Lecturer (2015–2019).