IEEE Open Journal of Vehicular Technology

Received 28 April 2021; accepted 20 May 2021. Date of publication 1 June 2021;
date of current version 23 June 2021. The review of this article was arranged by Guest Editor Dinh Thai Hoang.

Digital Object Identifier 10.1109/OJVT.2021.3085421

Deep Reinforcement Learning Based Energy Efficient Multi-UAV Data Collection for IoT Networks

SEYED SAEED KHODAPARAST, XIAO LU (Member, IEEE), PING WANG (Senior Member, IEEE), AND UYEN TRANG NGUYEN (Member, IEEE)

Department of Electrical Engineering and Computer Science, York University, Toronto, ON M3J 1P3, Canada

CORRESPONDING AUTHOR: PING WANG (e-mail: pingw@yorku.ca).

This work was supported by Canada NSERC Discovery under Grant RGPIN-2019-06375.

ABSTRACT Unmanned aerial vehicles (UAVs) are regarded as an emerging technology, which can be effectively utilized to perform the data collection tasks in the Internet of Things (IoT) networks. However, both the UAVs and the sensors in these networks are energy-limited devices, which necessitates an energy-efficient data collection procedure to ensure the network lifetime. In this paper, we propose a multi-UAV-assisted network, where the UAVs fly to the ground sensors and control the sensor's transmit power during the data collection time. Our goal is to minimize the total energy consumption of the UAVs and the sensors, which is needed to accomplish the data collection mission. We formulate this problem into three sub-problems of single UAV navigation, sensor power control as well as multi-UAV scheduling and model each part as a finite-horizon Markov Decision Process (MDP). We deploy deep reinforcement learning (DRL)-based frameworks to solve each part. Specifically, we use deep deterministic policy gradient (DDPG) method to generate the best trajectory for the UAVs in an obstacle-constraint environment, given its starting position and the target sensor. We also deploy DDPG to control the sensor's transmit power during data collection. To schedule activity plans for each UAV to visit the sensors, we propose a multi-agent deep Q-learning (DQL) approach by taking the total energy consumption of the UAVs on each path into account. Our simulations show that the UAVs can find a safe and optimal path for each of their trips. Continuous power control of the sensors achieves better performance over the fixed power approaches in terms of the total energy consumption during data collection. In addition, compared to the two commonly used baselines, our scheduling framework achieves better and near-optimal results.

INDEX TERMS Data collection, unmanned aerial vehicle (UAV), Internet of Things (IoT), deep reinforcement learning (DRL), energy consumption.

I. INTRODUCTION

Over the past few years, unmanned aerial vehicles (UAVs), commonly known as drones, have been increasingly used in a broad range of applications, including military services, surveillance and monitoring, telecommunications, and good's delivery [1], [2]. Due to their inherent desired features such as low cost, flexible maneuvering and ease of deployment [3], the UAVs can efficiently replace human operators in scenarios where it might be costly or hazardous.

With the recent advancements in the field of Internet-of-Things (IoT), wireless sensor networks (WSNs) have been widely deployed as a surveillance technology to monitor the surrounding environment, e.g., temperature and air pollution [4], [5]. These IoT sensors are usually energy-constrained devices with limited transmission ranges whose sensed data needs to be gathered and sent to the control center [6]. In that respect, UAVs, as mobile data collectors, can fly close to the dispersed sensors and collect the information. Compared to ground data collection schemes, UAVs are more efficient as they can easily adjust their path to reach a sensor by avoiding terrestrial obstacles [7]. In addition, due to their high altitude, the UAVs have a higher

chance of exploiting the Line-of-Sight (LoS) links to ground sensors [8].

However, despite all the advantages that UAVs have brought into communication networks, they have limited energy storage which poses a crucial challenge. After dispatching from the origin, they need to navigate towards the sensors while avoiding collisions with any obstacle. Also, using a single UAV to perform data collection from a large set of sensors distributed in a wide area may result in task failure. Collaborative multi-UAV approaches can be utilized to increase the chance of accomplishing the mission while reducing its completion time [9]. On the other hand, when the UAV arrives at the sensor's location and hovers above it to start the data collection, the communication link may experience different channel gains depending on the multi-path propagation environment. Hence, the sensor's transmission power needs to be controlled optimally to prolong its lifetime and provide reliable communication.

The complexity of the problem mentioned above makes it hard to be solved using traditional optimization techniques, which are usually time-consuming and need a complete model of the environment. Reinforcement learning (RL) allows us to find the solution by letting the agents, in our case, the UAVs, interact with the environment without knowing the model. In that way, the UAVs can make their own decisions after some trial and error during the training phase. Motivated by the above observations, we formulate the energy-efficient data collection problem as a finite-horizon Markov decision process (MDP). To overcome the computational complexity caused by the state space's high dimensionality, we propose a multi-UAV deep reinforcement learning (DRL)-based approach to minimize the energy consumption on both the UAVs and sensors sides. In particular, we develop and combine three different DRL frameworks for UAV's navigation, task scheduling, and the sensor's power control, respectively. We consider a delay-tolerant scenario where each sensor caches the sensed data and transmits it to a UAV upon the request [10].

The main contributions of this paper are summarized as follows:

1) We first propose an obstacle-aware navigation framework based on a deep deterministic policy gradient (DDPG) [11] method to help the UAV adjust its trajectory and speed from any given starting point towards its destination sensor with minimum energy consumption.
2) We propose a DDPG-based approach to control each sensor's transmit power while sending the data to the UAV. Unlike most of the related works, the transmit power of the sensor is not supposed to have discrete levels and can adaptively be changed depending on the multi-path propagation environment.
3) We propose a multi-UAV scheduling framework by incorporating the data provided by the navigation framework, with the aim to minimize the UAVs' overall energy consumption to accomplish the data collection, subject to their limited energy constraints. A deep Q-learning (DQL) [12] algorithm is developed to generate the activity plan of each UAV by providing a list of sensors that needs to be visited in order, considering the energy consumption between each two of them.
4) We conduct simulations to evaluate the performance of our proposed frameworks. Our simulation results show that the UAVs learn to navigate in the environment safely with a low collision rate. In addition, the power control model can successfully control the trade-off between the UAV's hovering and communication power and the sensor's transmit power, and the multi-UAV scheduling framework can achieve better performance comparing to the random and greedy baselines.

The remainder of this article is organized as follows: In Section II, we review the related works. Section III describes the system model for the considered scenario. Section IV presents the background of DRL. We propose our DRL frameworks utilized to solve the data collection problem in Section V. Section VI presents the simulation results for each of our frameworks. The conclusion of this paper is given in Section VII.

II. RELATED WORKS

The design of trajectory planning for UAV data collection has attracted increasing research attention recently. Existing approaches have exploited various optimization techniques. References [13], [14] focus on minimizing the task completion time in multi-UAV systems. Reference [13] solves the completion time minimization problem for a multi-UAV system by jointly optimizing the UAVs' data collection position and speed and sensors' transmit power, subject to the load requirement and energy constraint. The authors in [14] adopt a two-step approach that first optimizes the number and locations of cluster heads which aggregate the data from all sensors then determines the trajectories of the UAVs. Additionally, a heuristic genetic algorithm is designed to achieve a sub-optimal solution with low complexity. However, both [13] and [14] ignore the impact of small-scale fading.

Reference [6] investigates the UAV energy consumption minimization problem of data collection in a sensor network where the aggregated data of different cluster heads have heterogeneous deadlines. To solve the problem, the authors propose a bi-level solution that first optimizes the number and locations of the cluster heads based on K-means clustering and then minimizes the number of UAVs and the trajectory length by using a heuristic approach [15]. Different from [6], reference [16] aims to minimize the overall energy consumption of both the UAV and the sensors. To this end, the authors jointly optimize the data collection locations, the sensors to collect, and the UAV's trajectory under the constraints of the individual energy availability of the sensors. Different from the above works, which consider two-dimensional trajectory planning, the authors in [17] optimizes three-dimensional trajectory jointly with communication scheduling to maximize the minimum transmission rate of the sensors. The three-dimensional (3D) trajectory modelling allows the system to

incorporate more practical elevation angle-dependent Rician fading channels. However, the proposed approaches in [16] and [17] only apply for single-UAV systems.

The above-reviewed optimization techniques require the knowledge of complete system information in advance and can only perform data collection in a pre-scheduled manner. Hence, these approaches are not applicable in dynamic environments where instant system information may not be available. To handle data collection in a time-varying environment, research efforts have resorted to RL and DRL approaches which enable the UAVs to acquire solutions without knowing the complete system information. References [10], [18]–[22] investigate the data collection problem for single-UAV systems. The authors in [10] devise a Q-learning-based mechanism that achieves the optimal energy efficiency of the UAV, which needs to collect sensor data and return to a charging point before energy depletion during each charging circle. Reference [21] develops a double DQN-based mechanism to maximize the amount of collected data subject to completion time and navigation constraints. However, the approaches introduced in [10] and [21] only work for discrete flying actions. The focus of references [18]–[20], [22] is to minimize the age of information (AoI) of data collected from the distributed sensors. Reference [22] proposes an AoI-based trajectory planning algorithm for fresh data collection using a DRL technique. The authors in [19] introduce a DRL approach to optimize the UAV's trajectory and the communication scheduling of the sensors. However, the proposed algorithms do not take into account the energy consumption of the UAV. Reference [18] devises a DRL algorithm that achieves the same goals of [19] under the additional energy constraint of the UAV. The authors in [20] develop a DQN-based algorithm that minimizes the average AoI of the sensors while maintaining their packet drop rate as low as possible. However, neither [18] nor [20] considers the impact of the energy consumption of the sensors.

In addition to the above-reviewed designs for single-UAV systems, research efforts have also been devoted to addressing data collection in multi-UAV systems with DRL approaches. Reference [23] devises a sequential deep model for simultaneous task allocation and trajectory planning to maximize data collection ratio and geographic fairness with minimized total energy consumption of the UAVs. In [24], the authors propose a multi-agent DQL algorithm to maximize the minimum throughput by the joint optimization of path design and channel resource assignment in a UAV-enabled wireless powered communication network, where the UAVs can fly according to a discrete set of actions. References [25] and [26] aim to minimize the overall flight time of the UAVs under their individual energy constraints. The authors in [25] develop an option-based hybrid DRL method that allows the UAV to choose between two algorithms to handle deterministic and ambiguous boundary scenarios. However, the impact of obstacles on the navigation of the UAVs is ignored. Reference [26] addresses the obstacle-aware navigation based on a joint learning approach which first obtains the shortest route for data collection in an obstacle-constrained scenario based on DDPG and then determines the best schedule of the UAVs based on Q-learning. However, the proposed approach does not consider the power consumption of the sensors. Additionally, to our knowledge, none of the existing DRL-based data collection designs for multi-UAV systems takes into account the impact of small-scale fading. Being aware of the limitations, we aim to address the continuous trajectory planning for multi-UAV data collection in a practical obstacle-constrained environment with small-scale fading and power consumption of both UAVs and sensors taken into consideration.

III. SYSTEM MODEL

We consider a cooperative multi-UAV data collection task with N sensors randomly located on the region and M UAVs to collect the data gathered by the sensors. The position of UAV m and ground sensor n can be characterized by the coordinate (x_m, y_m, z_m) and $(x_n, y_n, 0)$, respectively. All the UAVs start from the origin (e.g., charging station), move towards the sensors, and go back to the origin when the data collection is finished. We assume that when a UAV arrives at the sensor's location, it hovers above the sensor and activates it by sending a query signal. During each communication round, the UAV can acquire the channel gain by exchanging signals and, based on that, controls the transmit power of the sensor. After the sensor has sent all its data, the UAV flies to the next target until all the data is collected.

A. ENVIRONMENT AND NAVIGATION MODEL

Since autonomous navigation in a real-world environment can be complex, we consider a 3D virtual environment with dense obstacles to match the complex urban areas. Our goal is to train a UAV to fly from any arbitrary starting location to any arbitrary destination avoiding collision with the obstacles and flying out of the borders of the environment.

For simplicity, we assume that the UAV flies at a fixed altitude h. Therefore, the position of UAV m in the 3D space is characterized by (x_m, y_m, h), and its movement is restricted to $x - y$ plane. The control profile of the UAV consists of its speed and orientation angle, and the UAV's dynamic at time t can be characterized by

$$\begin{aligned} x_m^{t+1} &= x_m^t + v_m^t \times cos(\phi_m^t) \\ y_m^{t+1} &= y_m^t + v_m^t \times sin(\phi_m^t), \end{aligned} \tag{1}$$

where v_m^t and ϕ_m^t are the UAV's speed and orientation angle, respectively.

Similar to [27], we assume that the UAV knows its current location and the destination by using GPS signals. To illustrate the UAV's perception of the surrounding environment, we assume that the UAV is equipped with range finders that can identify the distance from itself to obstacles in multiple directions. We know that if an obstacle's height is less than the UAV's altitude, it can simply be avoided and has no effect on the trajectory of the UAV. Without loss of generality, we

assume that all the obstacles in the environment have heights greater than the UAV's altitude.

B. UAV ENERGY CONSUMPTION MODEL

The energy consumption of the UAV includes two main components, namely the communication-related energy and propulsion energy. The communication-related energy is used in communication circuits of the UAV to send, receive and process the signals, which is negligible compared to the propulsion energy [28]. We assume the communication-related power is a constant denoted by P_c. The propulsion energy is needed to keep the UAV flying and hovering, which can be expressed as a function of its speed [29], i.e.,

$$P_{nav}(v) = P_0\left(1+\frac{3v^2}{U_{tip}^2}\right) + P_i\left(\sqrt{1+\frac{v^4}{4v_0^2}}-\frac{v^2}{2v_0^2}\right)^{\frac{1}{2}} + \frac{1}{2}d_0\rho\kappa A v^3, \tag{2}$$

where P_0 and P_i are constant parameters representing the blade profile power and induced power in hovering status, U_{tip} denotes the tip speed of the rotor blade, v_0 is the mean rotor-induced velocity during hovering. Moreover, the parameters d_0, κ, ρ and A represent the fuselage drag ratio, rotor solidity, air density and rotor disc area, respectively.

To obtain the power consumption during hovering, we let $v = 0$ in (2), which gives us the hovering power

$$P_h = P_0 + P_i. \tag{3}$$

We should note that as the UAV hovers above a sensor, it is using the communication-related power to receive the data. Therefore, the total power consumption in the data collection status is expressed as

$$P_{dc} = P_h + P_c. \tag{4}$$

C. CHANNEL MODEL AND DATA COLLECTION RATE

During data collection, the UAV hovers above the sensor at a high altitude, which increases the chance of establishing an LoS link. Hence, for the uplink ground-to-air (G2A) channel, we adopt the Rician fading channel consisting of a deterministic LoS link and a random multipath fading component, where the Rician factor is affected by the surrounding environment. The channel between UAV m and sensor n at time t can be modeled as [17]

$$h_t[m,n] = \sqrt{\beta[m,n]}g_t[m,n], \tag{5}$$

where $\beta[m,n]$ is the large-scale average channel power gain accounting for signal attenuation, including both the pathloss and shadowing, and $g_t[m,n]$ is the small-scale fading coefficient. Let $d_t[m,n]$ denote the distance between UAV m and sensor n which is given by

$$d_t[m,n] = \sqrt{(x_m-x_n)^2+(y_m-y_n)^2+h^2}. \tag{6}$$

We can express the average channel power gain $\beta_t[m,n]$ as

$$\beta_t[m,n] = \beta_0 d_t^{-\alpha}[m,n], \tag{7}$$

where β_0 is the average channel power gain at a reference distance of $d_0 = 1$ m, and α is the pathloss exponent that usually has a value between 2 and 6. Due to the existence of the LoS path, the small-scale coefficient follows the Rician distribution.

In the considered scenario, the UAV sends an activation signal to wake up the sensor. When the connection has been established, the UAV gets to know the channel gain from the sensor to the UAV based on the exchanged pilot signals. We assume that the UAV and ground sensors use advanced communication technology and are equipped with sectored antennas. Thus, we can write the achievable data rate as

$$K_t[m,n] = B\log\left(1+\frac{P_t G_t |h_t[m,n]|^2}{\sigma^2}\right), \tag{8}$$

where B, P_t, G_t, σ^2 are the bandwidth of the channel, transmit power of the sensor, the antenna power gain of the G2A link, and noise variance on the UAV side at time t, respectively. Since each UAV hovers above the sensor and the sensors are located far enough from each other, the impact of signal interference has not been considered in this paper. We use ζ_t to denote the available data to be collected at the sensor at time step t, which can be expressed as

$$\zeta_t^n = \zeta_{t-1}^n - T_s K_{t-1}[m,n], \tag{9}$$

where T_s is the time length of each time step. In addition, the initial data amount to be collected at each sensor is denoted by Λ, i.e., $\zeta_0^n = \Lambda$.

IV. PRELIMINARIES

In this section, we present the background of reinforcement learning frameworks used in the proposed solution. We introduce MDP as the framework to model our problem. Then, we explain the two DRL algorithms used to solve this problem.

In an MDP problem, an agent interacts with the environment by performing actions in discrete time steps. Specifically, at each time step t, the agent observes state s_t, takes an action a_t, receives a reward r_t based on s_t and a_t, and then goes to the next state s_{t+1}. The transition tuple is shown by (s_t, a_t, r_t, s_{t+1}). A policy $\pi(s)$ is defined as a mapping from state s to an action or a distribution over actions. In fact, the policy controls the actions of the agent, and therefore, the rewards it receives from the environment. We define the return R_t as the discounted sum of future rewards

$$R_t = \sum_{\tau=t}^{T}\gamma^{\tau-t}r_\tau, \tag{10}$$

where T is the final time step and γ is the discount factor ranging from 0 to 1. The discount factor determines the importance given to the immediate and future rewards. Higher values of γ indicate that the agent cares more about the future rewards, whereas the lower values show the opposite.

Our goal is to find a policy that maximizes the expected return from the start time step, which is expressed as

$$\pi^* = \arg\max_{\pi} \mathbb{E}[R_0|\pi]. \tag{11}$$

We define the action-value function $Q(\cdot)$ to approximate the expected value of R_t when starting from state s_t, taking action a_t, and then following the policy π:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}[R_t|s_t, a_t]. \tag{12}$$

The well-known algorithm in RL, Q-learning (QL), is a tabular method that aims to find the Q-values for each state-action pair (s, a), and after the training, follows the greedy action to achieve the optimal policy as expressed in

$$\pi(s) = \arg\max_{a} Q(s, a). \tag{13}$$

However, QL is not efficient when the state and action spaces are large, as we need to store the Q-value for each state-action pair in the table. DRL techniques can be utilized to overcome this challenge by combining deep neural networks (DNNs) and RL. Now, we explain the two DRL algorithms used in this paper, i.e., DQL and DDPG.

DQL makes use of a DNN to approximate the $Q(\cdot)$ function by minimizing the loss L

$$L(\theta^Q) = \mathbb{E}\left[(Q(s_t, a_t|\theta^Q) - y_t)^2\right], \tag{14}$$

where θ^Q denotes the weight vector of the DNN, and y_t known as target is expressed as

$$y_t = r(s_t, a_t) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}|\theta^Q). \tag{15}$$

In [12], Mnih *et al.* introduced two major changes i.e., experience replay and target network to resolve the issue of instability. To update the DNN, we use a mini-batch from an experience replay buffer with state transition tuples collected during learning (instead of the immediately collected sample). Compared to immediate sampling used in traditional QL, experience replay breaks correlations between sequentially generated samples, thus can avoid divergence and smooth out learning. Moreover, DQL uses an additional target network to estimate target values y_t for the training. A target network has the same structure as the original DNN. However, its weights are updated slowly with the original weights every a few epochs and are held fixed in between.

The problem with DQL is that it only works for control problems with a low-dimensional discrete action space. It is hard to apply DQL to continuous control because it needs to figure out the action that maximizes the $Q(\cdot)$ function in (13), which is quite difficult. The DQL-based method can only handle tasks with a limited discrete action space. However, UAV control is a continuous control task.

DDPG [11] as an actor-critic method was designed for this purpose. Both actor and critic are implemented by DNNs, where critic is acted as $Q(s_t, a_t|\theta^Q)$. Thus, it uses the same loss function as (14) for training. Actor uses a parameterized policy $\pi(s_t|\theta^{\pi})$ to generate an optimal action given state s_t. Instead of minimizing a loss function, the actor network maximizes the $Q(\cdot)$ in (12) by applying the gradient which can be written as

$$\nabla_{\theta^{\pi}} Q = \mathbb{E}\left[\nabla_a Q(s, a|\theta^Q)|_{a=\pi(s)} \nabla_{\theta^{\pi}} \pi(s|\theta^{\pi})\right]. \tag{16}$$

V. PROPOSED DRL FRAMEWORKS FOR AUTONOMOUS DATA COLLECTION

In this section, we propose the DRL-based framework for multi-UAV data collection problem. We divide the data collection problem into three parts: single UAV navigation, sensor power control and multi-UAV scheduling. We show how to model each part as an MDP and solve it with the aforementioned DRL methods.

A. MDP MODELING OF DATA COLLECTION PROBLEM

1) SINGLE UAV NAVIGATION

Suppose the UAV m is located at the start point (x_m^0, y_m^0, h), aiming to fly towards the sensor n located at $(x_n, y_n, 0)$. For the purpose of navigation, the UAV needs to use its sensory observation of the environment to avoid the obstacles, as well as its relative position to the sensor to reach the destination. The sensory observation of the UAV is obtained by using the range finders. The relative position to the sensor can be measured by GPS. Since the UAV flies at a fixed altitude, we just consider the variable axes, and hence, the relationship between the UAV and the sensor at time step t can be written as

$$(\psi_x^t, \psi_y^t) = (x_m^t, y_m^t) - (x_n, y_n). \tag{17}$$

We can denote the overall state of the UAV by $s_{nav}^t = (\psi_x^t, \psi_y^t, v_m^t, \phi_m^t, l_0^t, \ldots, l_4^t)$, where $l_0^t \sim l_4^t \in [0,\ 100]$ show the output of the range finders. The UAV can change its speed and orientation angle along the path, which can be expressed as

$$\begin{aligned} v_m^{t+1} &= v_m^t + \Delta v_m^t, \\ \phi_m^{t+1} &= \phi_m^t + \Delta \phi_m^t, \end{aligned} \tag{18}$$

where Δv_m^t and $\Delta \phi_m^t$ represent the throttle and steering signals. We denote the action of the UAV by $a_{nav}^t = (\Delta v_m^t, \Delta \phi_m^t)$. Note that both elements in the action vector are continuous variables.

The navigation task in an obstacle-constraint environment is complex. The reward function must be somehow designed to guide the UAV to reach its destination while satisfying the constraints. Our proposed reward function is composed of 4 parts: transition, energy consumption penalty, obstacle penalty, and finishing reward. The transition reward is designed as

$$r_{trans} = \lambda_1 \Delta d, \tag{19}$$

where λ_1 is a positive constant, and Δd shows the reduced distance to the sensor after taking the action. Δd is positive when the UAV becomes closer to the sensor, motivating the UAV to head for the sensor. We define the energy consumption

penalty as

$$r_e = -\lambda_2 P(v), \tag{20}$$

where λ_2 is a positive constant, and $P(v)$ is the power consumption of the UAV defined in (2). The UAV receives this penalty to adjust its speed and trajectory so that the energy consumption along the way is minimized. To encourage the UAV to avoid the obstacles, the obstacle penalty is designed as

$$r_{obs} = -\lambda_3 e^{-\lambda_4 l_{min}}, \tag{21}$$

where l_{min} is the minimum value among the range finders. If the UAVs becomes really close to an obstacle in either of the range finder directions, the penalty would increase exponentially. To further encourage the UAV to move towards the sensor, it would get a large constant positive reward r_{finish} when it arrives at the destination. The overall reward function for the navigation framework is formulated as follows

$$r_{nav} = r_{trans} + r_e + r_{obs} + r_{finish}. \tag{22}$$

When there are multiple UAVs working together for data collection, one UAV will consider other UAVs as obstacles if they fly at the same altitude. In this case, the aforementioned navigation approach can also be applied to avoid the collision among UAVs.

2) SENSOR POWER CONTROL

After arriving at the target sensor, the UAV hovers above it to gather the data. At each time step, the UAV sets a proper transmit power for the sensor. Since we have assumed the channel gain can be measured by the UAV at each time step, we take the channel gain into the state space to provide the UAV with better decisions. Low channel gain indicates that the sensor must transmit more power for a certain amount of data rate. Also, the amount of data remained at the sensor must be considered, since it helps the UAV achieve the goal state in which the remaining data must be zero. The state can be written as $s^t_{spc} = (g_t, \zeta_t)$ where g_t is the channel gain at time t, and ζ_t is the remaining data at the sensor. The action is simply denoted by $a^t_{spc} = P_t$, where P_t is the transmit power of the sensor controlled by the UAV.

For the reward function, we need to motivate the UAV to collect data, i.e., in this case having a high data rate. In addition, we must avoid using high transmission power in case the channel is not in a good condition. We should note that during data collection, the UAV itself is spending power for communication and hovering as stated in (4). With all these into consideration, we design the reward function as follows:

$$r_{spc} = \lambda_5 T_s K_t - \lambda_6 P_{dc} - \lambda_7 P_t, \tag{23}$$

where K_t is the data rate in (8), P_{dc} is the UAV's power consumption during data collection, and P_t is the transmission power of the sensor. λ_5, λ_6 and λ_7 are the coefficients that control the tradeoff between these three components.

3) MULTI-UAV SCHEDULING

The data collection is a task where the UAVs need to plan their destinations in a way that minimizes their navigation energy consumption. We utilize the scheduling framework to organize trip plans for each of the UAVs to visit the sensors. We assume that a sufficient number of UAVs have been deployed so that none of them runs out of battery to perform the cooperative data collection task. We also try to distribute the data collection task evenly to the UAVs to make the best use of their data storage capacity.

We consider the stop points for each UAV, namely the N sensors and the starting locations. We will denote the set of stop points by $\Omega = \{\omega_0, \omega_1, \ldots, \omega_N\}$, where ω_0 represents the starting point and the rest of them are the locations of the sensors. The cost of flying from point i to point j is equal to the UAV's energy consumption on that path, which can be obtained by using the navigation framework and is denoted by matrix $Cost(i, j)$. We normalize the matrix values by dividing them to the maximum value present in the matrix. Therefore, the maximum cost between two points is $+1$.

We denote the status of sensor i by $\nu(i) \in \{0, 1\}$, where the value of 1 states that the sensor has been visited by one of the UAVs. The state of UAV m can be written as $s^t_{sch,m} = (Pos(m), \nu(1), \ldots, \nu(N))$, where $Pos(m) \in \Omega$ is the UAV's current location, and $\nu(1), \ldots, \nu(N)$ denotes the status of all the sensors, which can be acquired by the help of a central controller or through the information exchange among UAVs. In this case, the action of UAV m is its choice for the next target location represented by $a^t_{sch,m} \in \Omega$.

We design the reward function to motivate the UAVs to find the best route starting from ω_0 and then coming back to it. The reward is expressed as

$$r^t_{sch,m} = \begin{cases} -1 & \text{if} a^t_{sch,m} = \omega_0 \text{ and} \\ & \exists \omega_i \in \Omega - \{\omega_0\} \\ & \text{where } \nu_(i) = 0 \\ -1 & \text{if} a^t_{sch,m} = Pos(m) \\ -Cost(Pos(m), a^t_{sch,m}) & \text{otherwise.} \end{cases} \tag{24}$$

In the first case, we penalize the UAV when it returns to ω_0, while there are unvisited sensors in the area. In the second case, the UAV is penalized when it stays at the current location, and hence, the UAV is motivated to fly between the sensors to collect data. The last case, is a regular case where the UAV gets a cost-related negative reward, when flying from sensor m to the next sensor indicated by the action $a^t_{sch,m}$.

Having individual rewards may benefit one, while punishing the others. However, the UAVs need to work cooperatively to schedule their visiting tour. Therefore, by sharing the same reward, we can ensure that they work as a team for the data collection. We define the overall reward at time step t as

$$r^t_{sch} = \sum_m r^t_{sch,m}. \tag{25}$$

Algorithm 1: DDPG Algorithm.

1: Randomly initialize critic $Q(s, a|\theta^Q)$ and actor $\pi(s|\theta^\pi)$ with weights θ^Q and θ^π
2: Initialize target network Q' and π' with weights $\theta^Q \leftarrow \theta^Q$ and $\theta^{\pi'} \leftarrow \theta^\pi$
3: Initialize the replay buffer W
4: **for** episode = 1,2,... **do**
5: Receive the initial observation state s_1
6: **for** t = 1,2,..., T_{max}^{DDPG} **do**
7: Select action $a_t = \pi(s_t|\theta^\pi) + \mathcal{N}_t$ according to the current policy and exploration noise
8: Execute action a_t, receive reward r_t and observe the new state s_{t+1}
9: Store transition (s_t, a_t, r_t, s_{t+1}) in W
10: Sample a random minibatch of I transitions (s_i, a_i, r_i, s_{i+1}) from W
11: Set $y_i = r(s_i, a_i) + \gamma Q'(s_{i+1}, \pi'(s_{i+1}|\theta^{\pi'})|\theta^{Q'})$
12: Update critic by minimizing the loss: $L(\theta^Q) = \frac{1}{I}\sum_i[(Q(s_i, a_i|\theta^Q) - y_i)^2]$
13: Update actor using the sampled policy gradient: $\nabla_{\theta^\pi} J \approx \frac{1}{I}\sum_i[\nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=a_i} \nabla_{\theta^\pi}\pi(s|\theta^\pi)|_{s_i}]$
14: Update the target networks: $\theta^{Q'} \leftarrow \eta\theta^Q + (1-\eta)\theta^{Q'}$ $\theta^{\pi'} \leftarrow \eta\theta^\pi + (1-\eta)\theta^{\pi'}$
15: **end for**
16: **end for**

B. DDPG TO SOLVE THE NAVIGATION AND SENSOR POWER CONTROL PROBLEM

To deal with continuous control problems of navigation and sensor power control, we use the well-known actor-critic architecture, DDPG, as the framework to solve our problem. DDPG algorithm is formally presented in Algorithm 1. The algorithm works as follows.

The weight vectors θ^Q and θ^π of the actor and critic networks are randomly initialized at the start of the algorithm (Line 1). As described earlier, we use target networks π' and Q' to improve learning stability. The target networks have the same structure as the original actor or critic networks. We initialize their weights in the same way as their original networks (Line 2). The target networks use soft updates, where they slowly track the original network weights (Line 14).

During the training process, the agent can interact with the environment to learn the optimal policy. The agent takes its actions according to the actor network and a noise process $\mathcal{N}_t$ (Line 7). We add the noise process to ensure the agent's exploration in the environment. Otherwise, we will not be able to try different actions, since the deterministic policy outputs just a single action. After taking the action, the agent receives a reward r_t and observes the next state s_{t+1}. We use a replay buffer to store the transition tuples (s_t, a_t, r_t, s_{t+1}) (Line 9). Then, a mini-batch is randomly sampled to train the actor and critic networks (Line 10). We update the critic network weights θ^Q by minimizing the loss function in (14). The sample-based version of the policy gradient is used to update the actor weights θ^π (Line 12,13).

Algorithm 2: Multi-Agent DQL Algorithm.

1: Randomly initialize critic $Q_{sch,m}(s, a|\theta^Q)$ with weights $\theta^{Q_{sch,m}}$ for all the UAV agents
2: Initialize target network $Q'_{sch,m}$ with weights $\theta^{Q'_{sch,m}} \leftarrow \theta^{Q_{sch,m}}$
3: Initialize the replay buffer W_m and ϵ for each UAV
4: **for** $episode = 1, 2, \ldots$ **do**
5: Receive the initial observation state s_1
6: **for** t = 1,2,..., T_{max}^{DQL} **do**
7: **for** m in {1,..., M} **do**
8: Select the action a_t^m for UAV m according to ϵ-greedy policy
9: **end for**
10: Execute the actions together, receive individual reward r_t^m and observe new state s_{t+1}^m for each agent
11: Compute the overall reward in (25) by summing over the individual rewards
12: **for** m in {1,..., M} **do**
13: Update the agent's reward and store the transition $(s_t^m, a_t^m, r_{total}, s_{t+1}^m)$ in W_m
14: Sample a random minibatch of I transitions (s_i, a_i, r_i, s_{i+1}) from W_m
15: Set $y_i = r_i + \gamma \max_{a_{i+1}} Q'_{sch,m}(s_{i+1}, a_{i+1}|\theta^{Q'})$
16: Update the $Q_{sch,m}$ by minimizing the loss: $L(\theta^{Q_{sch,m}}) = \frac{1}{I}\sum_i[(Q_{sch,m}(s_i, a_i|\theta^Q) - y_i)^2]$
17: **end for**
18: **end for**
19: Decay ϵ by a factor if $\epsilon > \epsilon_{stop}$
20: Update the target networks when $episode\%N_{upd} = 0$ $\theta^{Q'_{sch,m}} \leftarrow \theta^{Q_{sch,m}}$
21: **end for**

C. MULTI-AGENT DQL TO SOLVE MULTI-UAV SCHEDULING PROBLEM

We adopt the multi-agent DQL framework to find the best scheduling plan for the UAVs, where each of them has a separate Q-network that controls its policy. We have presented the multi-agent DQL solution in Algorithm 2. Our algorithm works as follows.

At the beginning, we initialize the network weights for each of the UAVs. We also do the same procedure for the target networks (Line 1,2). Unlike DDPG, the target networks are updated every N_{upd} episodes by fetching the original network weights (Line 20). As the training begins, in each time step, the agents choose their actions by using the ϵ-greedy policy (Line 8), where each agent chooses a random action with

TABLE 1 Parameters Used in the Paper

Parameter	Description	Simulation Value
f_c	Carrier frequency	2 GHz
B	Bandwidth	1 MHz
σ^2	Noise power	-100 dB
(ξ_1, ξ_2)	Rician distribution parameters	(0.5, 1)
α	Pathloss exponent	2
Λ	Data size of each sensor	100 Mbits
U_{tip}	Tip speed of the rotor blade	200
v_0	Mean rotor induced velocity in hovering	7.2
d_0	Fuselage drag ratio	0.3
ρ	Air density	1.225
s	Rotor solidity	0.05
A	Rotor disc area	0.79
P_0	Blade profile power in hovering	580.65
P_i	Induced power in hovering	790.67

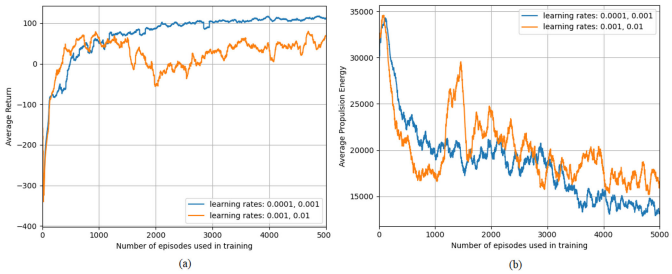

FIGURE 1. Convergence of the DDPG model during the training for the navigation task for different learning rates of actor and critic networks: (a) the average return received by the UAV in the last 100 episodes and (b) the average propulsion energy consumed by the UAV in the last 100 episodes.

probability ϵ, otherwise the greedy action $\arg\max_a Q(s, a)$ is selected based on the agent's Q-network. After receiving all the actions, the system generates the reward and the next state for each of the agents (Line 10). We should note that according to our formulation in (25), the updated reward is given to the agents by summing over their individual rewards (Line 11). Then, each agent stores its transition tuple in the replay buffer, samples a random tuple, and updates the network weights by minimizing the sample loss (Line 13-16). We decay ϵ in each episode to reduce the rate of exploration, as the agent begins to learn a better policy (Line 19).

VI. SIMULATION RESULTS

In this section, we discuss the experimental setting and the performance of the proposed frameworks. Our simulations are performed with Python 3.7 and Tensorflow 2.0 on an Intel Core i7-9700CPU with 16 GB RAM. The simulation parameters regarding the UAV's propulsion power and communication channel are shown in Table 1.

For our DDPG model, we use a 2-layer feedforward neural network architecture with 400 and 300 neurons in each layer for the actor and critic. Furthermore, the Adam optimizer [30] is used to update the network parameters, and similar to [11], learning rates of 10^{-4} and 10^{-3} are chosen for the actor and critic networks, respectively. In our implementation, $\mathcal{N}_t$ has normal distribution with mean of 0 and variance of 0.2 to explore the environment. We use minibatch sizes of 64, the discount factor is $\gamma = 0.99$, and the soft target update rate is $\eta = 0.001$. We use the same architecture and parameters as the critic network in DDPG for each of the agents in the multi-agent DQL model. ϵ starts from the value of 0.9 and decays by the factor of 0.999 until reaching $\epsilon_{stop} = 0.05$. We update the target networks every 10 episodes ($N_{upd} = 10$) to stabilize the learning process.

A. SINGLE UAV NAVIGATION

We simulate the obstacle-constraint environment, in which the UAV starts from a random location and is expected to fly to a destination. The target area is a 2D square of size 600×600 m^2 with obstacles randomly located in it. Here, we aim to reach the destination by avoiding the obstacles and minimizing the energy consumption of the UAV. The UAV's flight altitude h is set to 50 m. The reward is instantiated as: $\lambda_1 = 0.3$, $\lambda_2 = 0.002$, $\lambda_3 = 50$ and $\lambda_4 = 0.1$.

First, we illustrate the convergence of the proposed method. We trained the DDPG model for 5000 episodes, each of which has a maximum of 30 time steps. In Fig. 1, the convergence of the trained model with different learning rates of actor and critic networks is presented. Fig. 1(a) shows the average return formulated in (10) obtained by the UAV. Specifically, since we are randomly choosing the starting and finishing points in the area with different distances between them, the return received by the UAV in each episode can be different. Therefore, we average over the last 100 episodes to have a better view of the convergence of our model. We can see that the UAV learns to obtain the maximum accumulated reward in the obstacle-constrained environment with the original learning rates. In Fig. 1(b), we present the average propulsion energy consumed by the UAV in the last 100 episodes. We can see that as the model achieves a better return, the propulsion energy of the UAV decreases. The energy penalty defined in (20) causes the UAV to get higher rewards by decreasing the energy consumption on its path to the destination. Although the learning rates of 10^{-3} and 10^{-2} for the actor and critic networks perform better at the beginning, they result in a lower performance comparing to learning rates of 10^{-4} and 10^{-3}. Hereinafter, we set the learning rate as the latter ones.

In Figure 2, we present the trajectory followed by the trained UAV for 3 different environments, where the green circle shows the hovering location above the sensor, the red squares show the positions of the obstacles, and the blue dots indicate the trajectory of the UAV at each time step. We change the locations of obstacles, starting position and the destination and observe that the UAV is flexible to different scenarios, because we considered the relative position to the destination and the range finders for the navigation task. By adopting the continuous control actions, the UAV is able to change its path when getting close to an obstacle.

We also test the model for a period of 1000 episodes to validate the successful navigation without any collision to obstacles. The UAV reaches the target safely for 90.8% of the test episodes.

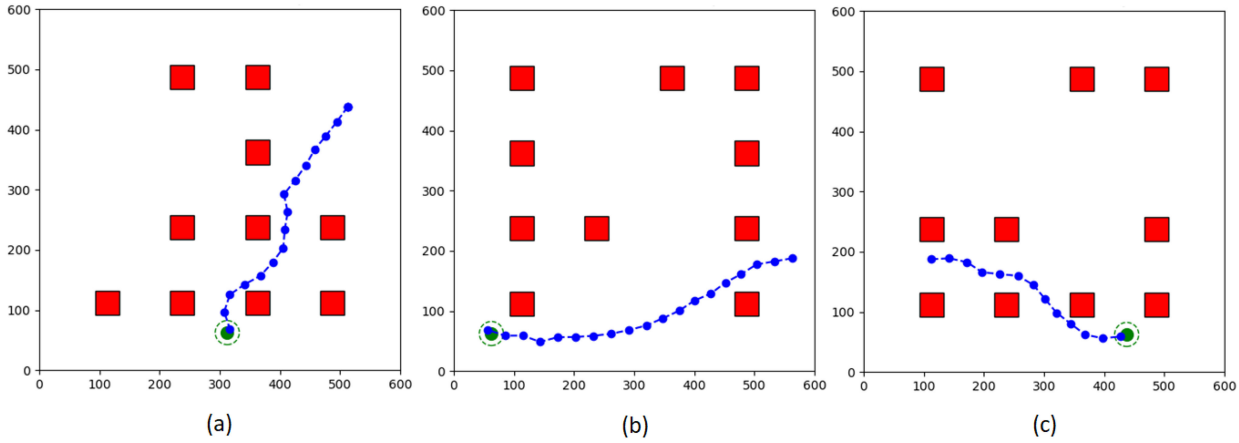

FIGURE 2. UAV's trajectory in three different environment configurations.

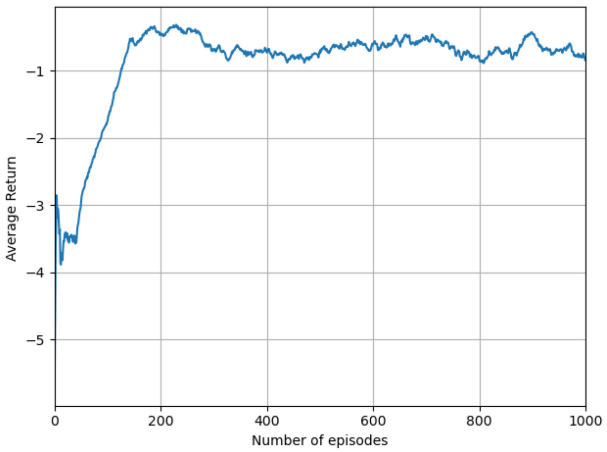

FIGURE 3. Convergence of the DDPG model during the training for the sensor power control.

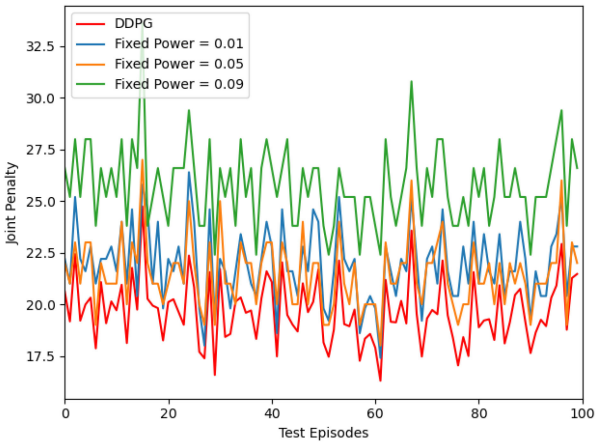

FIGURE 4. Joint penalty received by the UAV during the power control.

B. SENSOR POWER CONTROL

We consider the scenario where the UAV hovers above a sensor to collect its data. Considering the reward function in (23), our goal is to jointly optimize the UAV and sensor's power consumption during data collection. We investigate the impact of different coefficient, which can control the trade-off between the two types of power consumption. We assume the transmit power range of the sensor is between 0 and 0.1 W.

In Fig. 3, we illustrate the average return received by the agent during training. Since in each episode, the agent may experience different channel gains, we average over the last 100 episodes to smooth the curve. Compared to the navigation task, the DDPG model converges faster (about 10 times). This is particularly due to the fact that we have a much simpler state and action spaces, which speeds up the convergence. Our state consists of the values of channel gain and normalized remaining data, and our action is the transmit power of the sensor.

We compare the reward received by our adaptive model to some fixed power scenarios in the testing phase. Since the accumulated data collected by the UAV is constant in each episode and equal to 100 Mbits, the accumulation of the first term $\lambda_5 T_s R_t$ in the reward function (23) would be the same for all the scenarios. We added this term to the reward to motivate the UAV to collect data from the sensor. Hence, we take it out for comparison purposes and focus on the joint penalty $\lambda_6 P_{dc} + \lambda_7 P_t$ received by the agent. The penalty considers the power consumed by both the UAV and the sensor jointly at the same time. We choose the 3 values of 0.01 W, 0.05 W, 0.09 W for our fixed power scenarios. For ease of exposition, we present the results for 100 episodes. In all the scenarios, the environment experienced by the agent, i.e., the channel gains it receives in a specific episode and time step, is exactly the same. Fig. 4 shows the result for the case of $\lambda_6 P_{dc} = 0.5$ and $\lambda_7 = 10$. As we can see, the DDPG model achieves better performance against the fixed power approaches. We should note that as the transmit power of the sensor increases, we are likely to have a higher data rate, and the time required to collect all the data from the sensor decreases. As a consequence, the UAV spends a smaller amount of power for its communication and hovering. The model has learned the trade-off between P_{dc} and P_t successfully, and can adaptively adjust the sensor's power by considering the environment. For example, when we

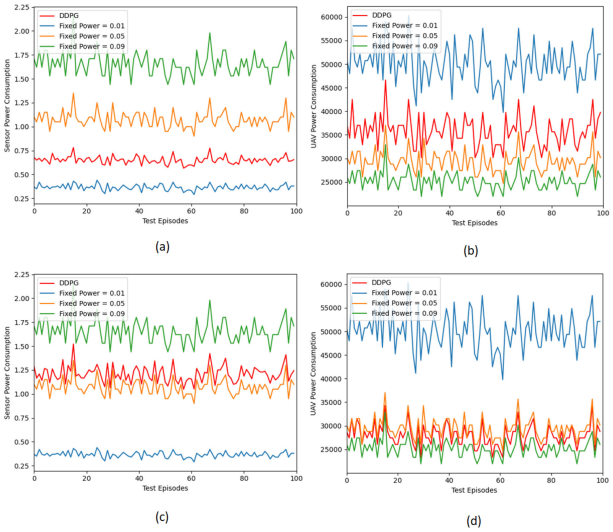

FIGURE 5. Impact of UAV power penalty coefficient on the learned policy: in (a) and (b) $\lambda_6 P_{dc} = 0.5$, and in (c) and (d) $\lambda_6 P_{dc} = 2$.

have a poor channel condition, it assigns a low power to the sensor to compensate for the later time when we have better channel conditions.

We also investigate the impact of coefficients on the learned policy and the trade-off between the UAV and the sensor's power consumption. Fig. 5 presents the results for two different cases. We keep the value of λ_7 fixed while changing $\lambda_6 P_{dc}$. In Fig. 5 (a) and (b) we consider $\lambda_6 P_{dc} = 0.5$, while in Fig. 5 (c) and (d) $\lambda_6 P_{dc} = 2$. We see that as we increase the penalty for the UAV's power, the agent learns to accomplish the task earlier, leading to lower UAV power consumption. However, the agent needs to increase the sensor's power to achieve such a goal, resulting in a higher sensor power consumption.

C. MULTI-UAV SCHEDULING

In this section, we study the behavior of the multi-agent scheduling framework in the selected scenario, shown in Fig. (6). The black spot at the bottom left shows the departure point for the UAVs (ω_0), the red squares show the positions of the obstacle, and the green circles are the sensor locations ($\omega_1 - \omega_6$). The UAVs start their trip from ω_0 and fly to the sensors to collect their data. For ease of illustration, we have considered 2 UAVs, however the presented framework is capable of handling a general case with more UAVs.

Fig. 7 depicts the convergence of the model in terms of the overall reward received by the agents in each episode of the training phase. As we can see, at the beginning episodes, the agents do not know how to interact with each other and receive a high amount of penalty, but as we progress in training, their policies improve, resulting in a lower penalty. We should mention that in this scheduling environment, the agents do not get any positive reward. The fluctuations appeared in the figure are the result of using ϵ-greedy policies, which makes the agents choose random actions during training to explore the effect of taking different actions.

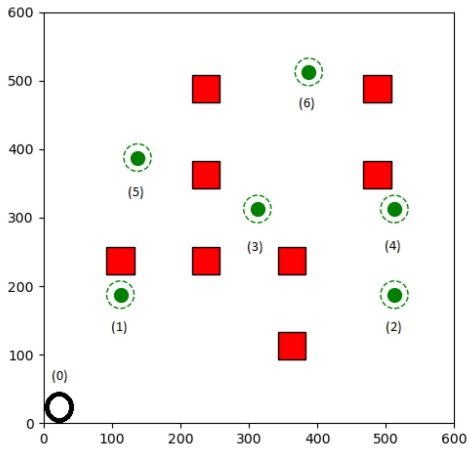

FIGURE 6. Considered scenario for data collection: the black spot shows the departure point of the UAVs (ω_0). The green circle illustrates the sensor locations ($\omega_1 - \omega_6$).

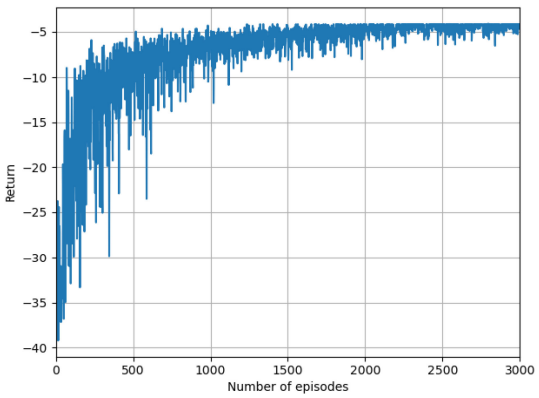

FIGURE 7. Convergence of the DQL model during the training in terms of the overall reward received by the agents.

TABLE 2 Scheduling Costs for Different Methods

Method	UAV 1	UAV 2	Cost
DQL	$0 \to 1 \to 5 \to 6 \to 0$	$0 \to 3 \to 4 \to 2 \to 0$	94900
Random	$0 \to 1 \to 3 \to 2 \to 0$	$0 \to 4 \to 6 \to 5 \to 0$	113926
Greedy	$0 \to 1 \to 3 \to 4 \to 0$	$0 \to 5 \to 6 \to 2 \to 0$	97953
Optimal	$0 \to 1 \to 5 \to 6 \to 0$	$0 \to 2 \to 4 \to 3 \to 0$	92785

We compare our proposed multi-agent approach to the random and greedy baselines. In the random approach, the agents choose the unvisited sensors randomly, whereas in the greedy approach, the agents fly immediately to the next unvisited sensor with the lowest energy consumption path. We also find the optimal solution by doing an exhaustive search. In all the solutions, the UAVs are required to collect data from the sensors evenly to make use of their storage capacity and decrease the data collection time. The results have been shown in Table 2. The 2nd and 3rd columns indicate the scheduled trip for UAV 1 and UAV 2 respectively, e.g., $0 \to 1 \to 5 \to 6 \to 0$ means that the UAV departs at ω_0, and then visits sensors ω_1, ω_5, ω_6, and finally goes back to the departure point ω_0. Our measure for this comparison is the total energy consumption of both UAVs when flying according to the scheduled trip, shown

as the cost in the last column of Table 2. We see that the DQL model achieves a closer result to the optimal solution comparing to the two other methods.

VII. CONCLUSION

In this work, we have proposed a DRL-based approach to solve the data collection problem for multiple UAVs in IoT networks with the aim to minimize the energy consumption on both the UAVs and sensors sides. We have divided the original problem into 3 sub-problems of navigation, sensor power control and multi-UAV scheduling and solved each sub-problem by utilizing a DRL algorithm. In the navigation framework, the DDPG algorithm has been deployed to enable each UAV to generate its trajectory autonomously by adjusting its speed and orientation angle, given the starting and finishing positions. The reward function has been designed to make the UAVs avoid the obstacles with the help of their sensory observations. For the sensor power control, we have used the DDPG to control the sensor's transmit power adaptively based on the current channel gain and the remaining data. To schedule the trip plans for each UAV, we have proposed the multi-agent DQL algorithm with the goal of minimizing the overall energy consumption of the UAVs during their flight on the scheduled paths. Our simulation results have shown that the UAVs can safely fly towards the target by avoiding the obstacles. Also, by controlling the sensor's power, we can obtain better performance than the fixed power approaches in terms of the total power consumption of the UAV and the sensor during the data collection time. Finally, our scheduling framework achieves a performance close to the optimal solution.

REFERENCES

[1] M. Mozaffari and M. Debbah, "A tutorial on uavs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2334–2360, Jul.–Sep. 2019.

[2] K. Kuru, D. Ansell, W. Khan, and H. Yetgin, "Analysis and optimization of unmanned aerial vehicle swarms in logistics: An intelligent delivery platform," *IEEE Access*, vol. 7, pp. 15 804–15 831, 2019.

[3] J. Wang, C. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Veh. Technol. Mag.*, vol. 12, no. 3, pp. 73–82, Sep. 2017.

[4] G. Ding, Q. Wu, L. Zhang, Y. Lin, T. A. Tsiftsis, and Y. Yao, "An amateur drone surveillance system based on the cognitive Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 29–35, Jan. 2018.

[5] A. Kadri *et al.*, "Wireless sensor network for real-time air pollution monitoring," in *Proc. 1st Int. Conf. Commun., Signal Process., Their Appl.*, 2013, pp. 1–5.

[6] O. Ghdiri *et al.*, "Energy-efficient multi-uav data collection for IoT networks with time deadlines," in *Proc. IEEE Glob. Commun. Conf.*, 2020.

[7] C. Y. Tazibt and *et al.*, "Wireless sensor network clustering for UAV-based data gathering," in *Proc. Wireless Days*, 2017, pp. 245–247.

[8] C. H. Liu, Z. Chen, J. Tang, J. Xu, and C. Piao, "Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 2059–2070, Sep. 2018.

[9] Y. Wang *et al.*, "Multi-UAV collaborative data collection for IoT devices powered by battery," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2020, pp. 1–6.

[10] S. Fu *et al.*, "Energy-efficient UAV enabled data collection via wireless charging: A reinforcement learning approach," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2021.3051370.

[11] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2016.

[12] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[13] Y. Wang, Z. Hu, X. Wen, Z. Lu, and J. Miao, "Minimizing data collection time with collaborative UAVs in wireless sensor networks," *IEEE Access*, vol. 8, pp. 98 659–98 669, 2020.

[14] S. Alfattani, W. Jaafar, H. Yanikomeroglu, and A. Yongacoglu, "Multi-uav data collection framework for wireless sensor networks," in *Proc. IEEE Glob. Commun. Conf.*, 2019, pp. 1–6.

[15] J. K. Lenstra and A. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[16] M. B. Ghorbel, D. Rodríguez-Duarte, H. Ghazzai, M. J. Hossain, and H. Menouar, "Joint position and travel path optimization for energy efficient wireless data gathering using unmanned aerial vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2165–2175, Mar. 2019.

[17] C. You and R. Zhang, "3D trajectory optimization in rician fading for uav-enabled data harvesting," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3192–3207, Jun. 2019.

[18] M. Yi, X. Wang, J. Liu, Y. Zhang, and B. Bai, "Deep reinforcement learning for fresh data collection in UAV-assisted IoT networks," in *Proc. IEEE INFOCOM 2020-IEEE Conf. Comput. Commun. Workshops*, 2020, pp. 716–721.

[19] M. A. Abd-Elmagid, A. Ferdowsi, H. S. Dhillon, and W. Saad, "Deep reinforcement learning for minimizing age-of-information in UAV-assisted Networks," in *Proc. IEEE Glob. Commun. Conf.*, 2019, pp. 1–6.

[20] P. Tong, J. Liu, X. Wang, B. Bai, and H. Dai, "Deep reinforcement learning for efficient data collection in UAV-aided Internet of Things," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2020, pp. 1–6.

[21] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "UAV path planning for wireless data harvesting: A deep reinforcement learning approach," in *Proc. IEEE Glob. Commun. Conf.*, 2020.

[22] C. Zhou *et al.*, "Deep RL-based trajectory planning for AOI minimization in UAV-assisted IoT," in *Proc. IEEE 11th Int. Conf. Wireless Commun. Signal Process.*, 2019, pp. 1–6.

[23] C. Piao and C. H. Liu, "Energy-efficient mobile crowdsensing by unmanned vehicles: A sequential deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6312–6324, Jul. 2020.

[24] J. Tang, J. Song, J. Ou, J. Luo, X. Zhang, and K.-K. Wong, "Minimum throughput maximization for multi-UAV enabled WPCN: A deep reinforcement learning method," *IEEE Access*, vol. 8, pp. 9124–9132, 2020.

[25] Y. Zhang, Z. Mou, F. Gao, L. Xing, J. Jiang, and Z. Han, "Hierarchical deep reinforcement learning for backscattering data collection with multiple UAVs," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3786–3800, Mar. 2021.

[26] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "A UAV-assisted data collection for wireless sensor networks: Autonomous navigation and scheduling," *IEEE Access*, vol. 8, pp. 110 446–110 460, 2020.

[27] C. Wang, J. Wang, Y. Shen, and X. Zhang, "Autonomous navigation of UAVs in large-scale complex environments: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2124–2136, Mar. 2019.

[28] C. Zhan and Y. Zeng, "Aerial-ground cost tradeoff for multi-UAV-enabled data collection in wireless sensor networks," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1937-1950, Mar. 2020.

[29] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing uav," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2015.

SEYED SAEED KHODAPARAST received the B.Sc. degree in electrical engineering from the University of Tehran, Tehran, Iran, in 2019. He is currently working toward the M.A.Sc. degree with the Lassonde School of Engineering, York University, Toronto, ON, Canada. His current research interests include deep reinforcement learning, autonomous vehicles, and the Internet-of-Things.

XIAO LU (Member, IEEE) received the B.Eng. degree from the Beijing University of Posts and Telecommunications, Beijing, China, the M.Eng. degree from Nanyang Technological University, Singapore, and the Ph.D. degree from the University of Alberta, Edmonton, AB, Canada. His current research interests include design, analysis, and optimization of future generation wireless networks.

PING WANG (Senior Member, IEEE) received the bachelor's and master's degrees in electrical and computer engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2008. In August 2018, she joined York University, Toronto, ON, Canada, as an Associate Professor. Prior to that, she worked with Nanyang Technological University, Singapore, from 2008 to July 2018. Her main research interests include wireless communication networks, cloud computing, and the Internet of Things. Her scholarly works have been widely disseminated through top-ranked IEEE journals/conferences and was the recipient of the Best Paper Awards from IEEE Wireless Communications and Networking Conference (WCNC) in 2012 and 2020, from IEEE Communication Society: Green Communications and Computing Technical Committee in 2018, and from IEEE International Conference on Communications (ICC) in 2007.

UYEN TRANG NGUYEN (Member, IEEE) received the B.Sc. and M.Sc. degrees from Concordia University, Montreal, QC, Canada, and the Ph.D. degree in computer science from the University of Toronto, Toronto, ON, Canada. She is currently an Associate Professor with the Lassonde School of Engineering, York University, Toronto, ON, Canada. Her research interests include wireless networking, mobile computing, online social networking, and information security. Her research has been funded by NSERC (National Sciences & Engineering Research Council of Canada), MITACS (Mathematics of Information Technology and Complex Systems), York University and industry partners. She is a frequent reviewer of grant applications for NSERC, MITACS, the U.S. National Science Foundation, and the National Science Centre of Poland.