applied sciences

MDPI

Article

The UAV Trajectory Optimization for Data Collection from Time-Constrained IoT Devices: A Hierarchical Deep Q-Network Approach

Zhenquan Qin [1,2], Xuan Zhang [1], Xinwei Zhang [1], Bingxian Lu [1,2,*], Zhonghao Liu [1] and Linlin Guo [3]

[1] School of Software Technology, Dalian University of Technology, Dalian 116024, China; qzq@dlut.edu.cn (Z.Q.); zxuan@mail.dlut.edu.cn (X.Z.); zhangxinwei029@gmail.com (X.Z.); liuzhonghao@mail.dlut.edu.cn (Z.L.)
[2] Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian 116023, China
[3] School Information Science and Engineering, Shandong Normal University, Jinan 250220, China; linlin.teresa.guo@gmail.com
* Correspondence: bingxian.lu@dlut.edu.cn

Abstract: Recently, using unmanned aerial vehicles (UAVs) to collect information from distributed sensors has become one of the hotspots in the Internet of Things (IoT) research. However, previous studies on the UAV-assisted data acquisition systems focused mainly on shortening the acquisition time, reducing the energy consumption, and increasing the amount of collected data, but it lacked the optimization of data freshness. Moreover, we hope that UAVs can perform long-term data collection tasks in dynamic scenarios within a constantly changing age of information (AoI) and within their own power levels. Therefore, we aim to maximize the quality of service (QoS) based on the freshness of data, while considering the endurance of the UAVs. Since our scenario is not an inertial order decision process with uniform time slots, we first transform the optimization problem into a semi-Markov decision process (SMDP) through modeling, and then we propose a hierarchical deep Q-network (DQN)-based path-planning algorithm to learn the optimal strategy. The simulation results show that the algorithm is better than the benchmark algorithm, and the tradeoff between the system QoS and the safe power state can be achieved by adjusting the parameter β_e.

Keywords: UAV; QoS; SMDP; hierarchical DQN; AoI; trajectory planning

Citation: Qin, Z.; Zhang, X.; Zhang, X.; Lu, B.; Liu, Z.; Guo, L. The UAV Trajectory Optimization for Data Collection from Time-Constrained IoT Devices: A Hierarchical Deep Q-Network Approach. *Appl. Sci.* **2022**, *12*, 2546. https://doi.org/10.3390/app12052546

Academic Editors: Jenhui Chen, Lei Wang and Zhiqun Hu

Received: 6 January 2022
Accepted: 25 February 2022
Published: 28 February 2022

1. Introduction

Unmanned aerial vehicles (UAVs) have received extensive attention in wireless communication networks due to their small size, flexibility, mobility, and controllability [1–4]. One of the most important applications is using UAVs as flying base stations to collect data from ground nodes, especially in the area of the Internet of Things (IoT). Using UAVs to assist data collectio, can not only save energy by reducing communication distances and hops, but it also enables data collection tasks to be performed in areas lacking communication coverage. In [5–7], UAVs are used to transmit data for the ground nodes, and the trajectory planning goal is set to balance the transmission delay of the ground nodes, as well as reducing the energy consumption of the UAVs. However, UAVs consume additional propulsive energy during movement. Thus, a tradeoff is required between the transmission energy consumption of the ground nodes and the propulsion energy consumption of the UAVs [8]. Because of the limited capacity and coverage of cellular networks, it is challenging to accommodate massive IoT devices. UAVs can be used to provide additional radio access services [9]. Moreover, UAVs can fly close to each ground node and can establish air-to-ground channels dominated by their line of sight (LoS), which can greatly reduce the transmission energy consumption, and can significantly improve the throughput of the ground nodes. These advantages have made UAV-assisted IoT networks attract extensive

attention in recent years. A study by Abdulla et al. [10] demonstrated how to maximize the acquisition efficiency of the unmanned aircraft system (UAS), modeling the problem as a potential game between UAS and sensor nodes, and proposing a data acquisition method that maximizes energy efficiency under fairness constraints. Zhan et al. [11] considered jointly optimizing the wake-up time of the sensor nodes and the trajectory of UAVs. While ensuring the required amount of data is collected, the energy consumption is minimized, and the suboptimal solution is obtained by using the iterative convex optimization technique. Wang et al. [12] considered minimizing the completion time of UAV acquisition tasks, which builds on their previous research on joint optimization. Moataz et al. [13] considered the problem of UAV-assisted data acquisition in delay-sensitive application scenarios, and they maximized the available quantity of devices serving the IoT by optimizing the trajectory and resource allocation of UAVs.

Most of the existing works of UAV-assisted data acquisition systems focus mainly on shortening the acquisition time, reducing energy consumption, and increasing the amount of collected data, which is aimed at either maximizing system throughput or minimizing delay via designing the trajectory, deployment position, or resource allocation of UAVs [14–18]. However, for real-time IoT applications, such as air quality monitoring, precision agriculture, and post-disaster early warning, the quality of service (QoS) highly depends on the freshness of information collected by UAVs from the ground nodes. In this paper, we introduce the age of information (AoI) to indicate the freshness of collected data. Different from the delay metric, the AoI is defined as the time elapsed since the latest received status update packet at a destination node [19]. The continuous data collection task is also a research hotspot. Due to its learning ability and generalization ability, deep reinforcement learning (DRL) is very suitable for such dynamic scenarios in which the AoI is constantly changing [20,21]. In [15,22], the authors formulated the fresh data collection problem in UAV-assisted IoT networks as a Markov decision process (MDP) and used a DRL-based approach to minimize the AoI. Zhou et al. [23] considered the unknown and time-varying traffic generation pattern of ground nodes, and proposed a novel online AoI-based trajectory planning approach using a deep deterministic policy gradient (DDPG). However, the energy of the UAV is limited, and each UAV needs to recharge in continuous data collection tasks. None of these works take recharging the UAV into account. In addition, the MDP-based UAV path-planning method requires the environment to be divided into discrete areas with separate action spaces, which results in a lot of additional propulsion power consumption. To solve these problems, we consider, in our work, that UAVs can perform long-term data collection tasks in dynamic scenarios where the AoI and their own power levels are constantly changing. Therefore, we formulate the trajectory design problem of the UAVs as a semi-MDP (SMDP) and use a hierarchical deep Q-network (DQN)-based method. The contributions can be summarized as follows:

- We study the path-planning of rechargeable UAVs in continuous data collection tasks, in which a UAV moves towards the ground nodes to collect data during a period of time and returns to recharge at the right time. We introduce the AoI to indicate the freshness of the collected data;
- To optimize the freshness of collected data and to ensure that the remaining charge of the UAVs is sufficient, we formulate the path-planning problem as an SMDP and propose a hierarchical DQN (H-DQN) method to deal with it. The agent learns to select options at a high level and the action depends on the selected option.

The rest of the paper is organized as follows. We review related work regarding the AoI and the typical algorithm of DRL in Section 2. Then, we describe our system model in Section 3. In Section 4, the proposed algorithm is explained in detail. Afterwards, we present simulation experiments and analyses to validate the performance and efficiency of our proposed approach in Section 5. Moreover, we compare our proposed method with the baseline methods to show its advantages. Finally, we give the conclusions in Section 6.

2. Related Work

Traditional data communications pay attention to the time delay of information transmission. However, after the data is generated, the staleness of the data information will increase with the one-way passage of time. So, real-time update applications need to pay more attention to the freshness of information. The concept of the age of information (AoI) is for scenarios that require a high freshness of collected data, and it is used to describe the freshness of the data collected by the system [24,25]. The biggest difference between the AoI and time delay is that the AoI includes not only the transmission time delay of the information, but also the waiting time of the information at the source node and the residence time at the destination node. The AoI is defined as the difference between the current time and the generation time of the latest packet available at the receiving end. Both the average age and the average age distribution can be used to characterize the freshness of the information. There are some studies that have already explored the data freshness in the UAV-assisted IoT network for data collection. Cao et al. [26] considered a scenario where a UAV aims at a source node to transmit data packets to a remote destination node, minimizing the peak AoI by jointly optimizing the flight trajectory and power allocation. Yi et al. [27] proposed a dynamic programming approach to minimize the AoI at the data center by redesigning the path-planning of UAVs. In these works, the UAV collects the data from each ground node only once, and then flies back to the depot.

Reinforcement learning is a learning mechanism that obtains the long-term maximum reward by learning the mapping relationship between the environment state space and the agent action space [28]. Reinforcement learning can be divided into two categories: value-based and policy-based. Value-based methods mainly use the state-value function V and the state-action value function Q for policy selection, where the expressions of V and Q are in a recursive form. In the case of limited state and action space, all strategies can be evaluated to obtain the optimal strategy pi^*, but that is difficult to achieve in reality. A feasible method is to continuously optimize the policy through iterations. Firstly, this involves randomly initializing the value function of a policy, then selecting the action according to the value function and updating it according to the Bellman optimal equation value function until its convergence. Q-Learning is a common algorithm of value-based functions. However, traditional reinforcement learning methods can only be applied in some very simple scenarios due to the difficulty of convergence, the curse of dimensionality, and the low generalization. Deep learning is very suitable for regression problems. Using artificial neural networks to fit value functions can not only overcome the problem of a dimensional explosion of state–action space, but it can also give reinforcement learning a certain generalization ability, which makes deep reinforcement learning possible [29]. The introduction of deep learning into reinforcement learning also brings some new challenges. First, in deep learning, the samples used for training are often independent and identically distributed, while decision-making is inertial in reinforcement learning. There is a correlation between samples, which makes it difficult for the neural network to converge. Besides, in the original Q-Learning, the target values and the current evaluation values, as the sample labels, are generated by the same Q-table. However, in deep reinforcement learning, this approach causes the model to oscillate and diverge, so the network that generates the labels is constantly changing.

In order to solve the above problems, the deep Q-network (DQN) came into being [30]. The DQN is a classic representative of value-based algorithms. There is only one value function network in this algorithm, without a policy network. In [31], the authors believe that its biggest feature is that it can directly use the original high-dimensional sensor data for reinforcement learning, and can use the stochastic gradient descent for training, making the training process more stable. The DQN uses the experience reply mechanis; that is, it constructs a memory called the experience playback pool to store the samples collected each time, and it then removes the independent and identically-distributed samples through the random sampling for training, breaking the association of the data. For the second problem, a network identical to the Q-network is constructed in the DQN, called the target

Q-network. The network is updated after a certain training step, and the updated method involves copying the parameters in the Q-network. This method can reduce the possibility of oscillation divergence during training, and can make training more stable.

Liu et al. [17] proposed a QoS-based action selection strategy based on the double deep Q-network algorithm. UAVs can make real-time moving and unloading decisions according to the current system state. Wang et al. [32] transformed the path-planning problem of the UAV base stations into a constrained Markov decision process (CMDP), and used Q-Learning to solve the problem. Due to the large state dimension, an adversarial deep Q-Network (dueling deep Q-network) is proposed, which introduces neural networks and adversarial architectures. Yi et al. [33] considered throughput and user fairness, a deep Q-network based learning method is proposed to maximize the overall network utility in the downlink of heterogeneous networks. However, the current research rarely conducts a unified study of the UAV's energy consumption and the AoI-based quality of service.

In this paper, we propose an improved method based on the DQN algorithm, and we use it in our study.

3. System Model

3.1. Scenario

As shown in Figure 1, we consider a UAV-assisted post-disaster early warning scenario with a set $\mathcal{N}$ of N ground nodes randomly distributed on the ground, a hovering UAV, and a ground control station (GCS). The ground nodes are responsible for sensing the environmental information data of the target area, and the UAV takes off from the GCS and collects the data from each ground node. During the flying and collection process, if the remaining energy is insufficient, the UAV should return to the control station for recharging to ensure the completion of the mission. For the ease of deployment, we assume that the UAV flies at a fixed flight velocity v_u and a fixed altitude h. In addition, we require the UAV to fly directly above the ground node to collect data. Therefore, the UAV has $N+1$ predetermined waypoints, which are denoted by the set $\mathcal{P}^{\sqcap} = \{p_0^u, p_1^u, ..., p_N^u\}$, where p_0^u is the position of GCS and $p_n^u, n \in \mathcal{N}$ is the position of the n-th ground node. The trajectory of the UAV is approximated by the sequence $p^u = \{p_0^u, ..., p^u(i), ..., p_0^u\}$, where $p^u(i)$, $p^u(i) \in \mathcal{P}^{\sqcap}$ denotes the hover point of the i-th decision.

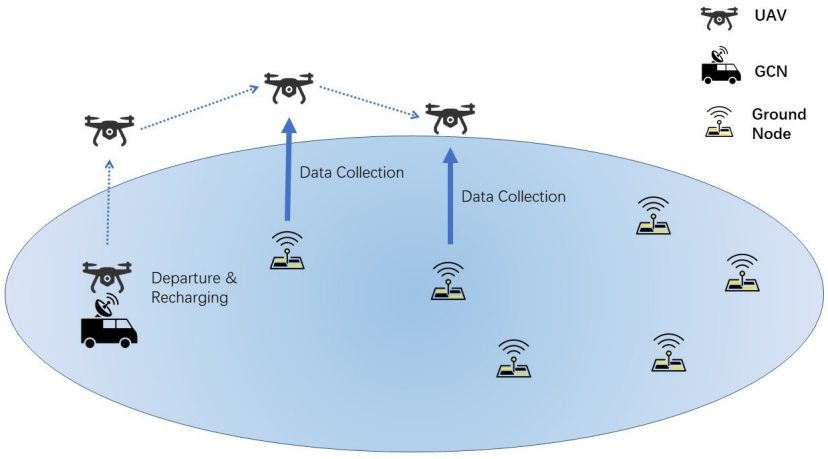

Figure 1. The UAV-assisted data collection scenario.

3.2. Communication Model

Similar to [27], the channels between the UAV and the ground nodes are assumed to be dominated by the LoS links. Therefore, the channel power gain between the UAV and the m-th ground node can be given by

$$g = \beta_0 d^{-2} = \frac{\beta_0}{h^2}, \tag{1}$$

where β_0 is the channel gain at a reference distance of 1 m, and d is the distance between the UAV and the ground sensing device. Since this paper assumes that the UAV collects data directly above the sensing device, $d = h$. Then, the uplink throughput between the UAV and the m-th ground node can be calculated as

$$R_n = Blog_2(1 + \gamma_n), \tag{2}$$

where B is the channel bandwidth, $\gamma_n = \frac{p_n g}{\sigma^2}$ is the signal-to-noise ratio (SNR) between the UAV and the n-th ground node, p_n is the transmitting power of the n-th ground node, and σ^2 denotes the Gaussian white noise power perceived at the UAV. The flight time of the i-th decision is then given by

$$T_f^i = \frac{\|p^u(i) - p^u(i-1)\|}{v_u}, \tag{3}$$

where v^u is the flying speed of the UAV. This paper assumes that the UAV is flying at a constant speed. The collecting time of the i-th decision can be given by

$$T_c^i = \frac{s_n}{R_n}, \tag{4}$$

where s_n is the size of the data generated by the n-th ground node.

3.3. Energy Model

The energy consumption of the UAVs consists of the communication energy and the propulsion energy. Since the communication energy consumption is relatively small, we only consider the propulsion energy in this paper. According to [34], the propulsion energy is used for keeping the UAV hovering, flying, and accelerating, which is related to the speed of the UAV. The propulsion energy of the rotary-wing UAV can be expressed as

$$\begin{aligned} E_u =& P_0(1 + \frac{3V_t^2}{U_{tip}^2}) + P_1(\sqrt{1 + \frac{V_t^4}{4v_0^4}} - \frac{V_t^2}{2v_0^2})^{\frac{1}{2}} \\ &+ \frac{1}{2} d_0 \rho s_0 A_r V_t^3, \end{aligned} \tag{5}$$

where P_0 and P_1 represent the blade profile power and derived power, respectively, V_t is the flying speed of the UAV in time slot t, U_{tip} is the tip speed of the rotor blade of the UAV, v_0 is the mean rotor induced velocity in the hovering state, d_0 is the fuselage drag ratio, ρ is the density of air, s_0 is the rotor solidity, and A_r represents the area of the rotor disk. In particular, the energy consumption, when hovering, is $E_u = P_0 + P_1$. Therefore, the energy consumption generated of the i-th decision can be expressed as

$$E_i = E(v)T_f^i + E(0)T_c^i, \tag{6}$$

where $E(v)$ is the UAV flight energy consumption and $E(0)$ is the hovering energy consumption. In this paper, it is assumed that E_max is the initial energy consumption of the UAV.

3.4. Freshness of Collected Data

We employ the AoI to measure the freshness of collected data, which is defined as the time that elapsed since the generation of the latest status update received by the UAV. The AoI of the n-th ground node at time slot t can be expressed by

$$\Delta_n(t) = t - U_n(t), \tag{7}$$

where $U_n(t)$ denotes the time at which the latest status update of the n-th ground node, successfully received by the UAV, was generated.

Figure 2 describes the characteristics of the AoI. It can be seen that when there is no data collection action, the AoI of the sensing device increases in a stepwise manner. When data is successfully collected, the AoI will decrease to the time it takes to generate the latest data. Moreover, the data freshness should be evaluated based on how valuable the fresh data is in the specific application. Therefore, data freshness can be characterized by a non-increasing utility function. Our optimization objective is the AoI-based system utility. The AoI-based system utility at the time slot t can be expressed as

$$Q(t) = \frac{1}{N} \sum_{n=1}^{N} u_n(\Delta_n(t)), \tag{8}$$

according to [35], we define the function $u(\Delta) = a^{w\Delta}$, where $0 < a < 1$ is a constant, and w is the time-sensitive weight.

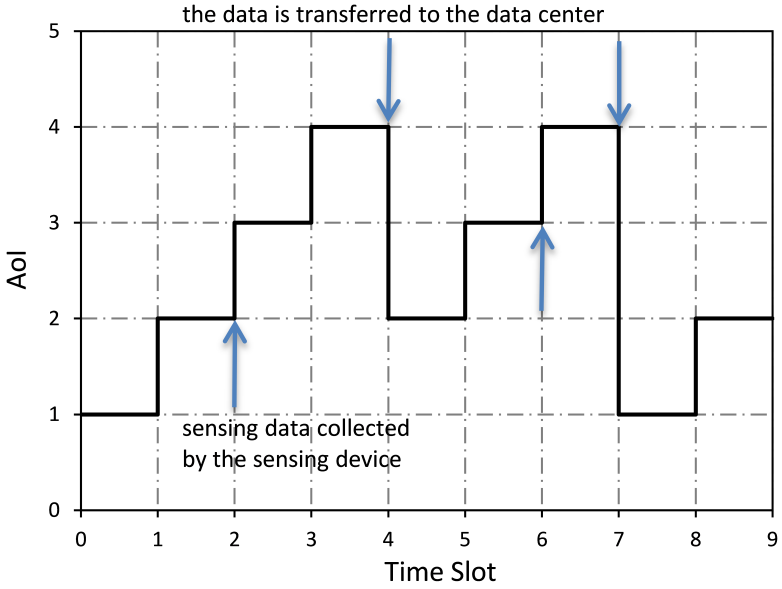

Figure 2. Example diagram of AoI changes.

3.5. Optimization Problem

We consider a discrete-time system, where the entire time is discretized into the t time slots of equal length, denoted as $\mathcal{T} = \{1, 2, \ldots, T\}$, and the unit length is denoted as δ. On the basis of equal-length time slots, the time system is abstracted into multiple execution cycles of unequal length. The duration of one cycle is the time it takes to execute a decision, and the length of each time slot depends on the task performed in this time slot. For example, in the y-th cycle, the UAV performs the data acquisition task of the n-th device, $n \in N$, then the execution time is $T^i = T^i_f + T^i_c$, where T^i_f represents the flight time to the target sensing device in (3), and T^i_c represents the time of the data transmission in (4).

According to the model and formulas described above, the central goal of the multi-UAV path-planning problem can be modeled as the following maximization problem

$$\max_{p^u} \sum_{i=1}^{I} \sum_{t=\sum_{j=1}^{i-1} T^j}^{\sum_{j=1}^{i} T^j} Q(t), \tag{9}$$

$$s.t. \quad p^u_i \in \{p_1, ..., p_N\}, \forall i \in \mathcal{I}, \tag{10}$$

$$e_i > 0, \forall i \in \mathcal{I}. \tag{11}$$

The first constraint in (10) indicates that the target point of the UAV can only be selected from the set of specified locations for each decision. The second constraint in (11) indicates that the energy of the UAV during the mission will not be exhausted prematurely, where e_i stands for the remaining energy.

4. Proposed Trajectory Optimization Approach

Different from common reinforcement learning problems, the scenario modeled in this paper is not an inertial order decision process with evenly divided time slots. Therefore, a time abstraction strategy is adopted, using a semi-Markov decision process (SMDP) to describe this problem. At first, we abstract the UAV as an agent, and build the system model as an SMDP problem by designing the system state space, the action space, and the reward function. Then, a path-planning algorithm based on the hierarchical DQN is proposed, which enables the UAVs to perform long-term data collection tasks in dynamic scenarios with constantly changing AoI and their own power.

4.1. SMDP

An MDP is often used to simplify the modeling of scenarios in reinforcement learning. It assumes that the state transition of the environment has Markov properties; that is, the probability of transitioning to the next state is only related to the previous state. The agent continuously observes the environment with Markov characteristics and makes habitual decisions in an MDP. An MDP with M agents consists of a quadruple $\langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{S}$ represents the state space of the environment, which is used to describe the state of each entity. In addition, $\mathcal{O}$ represents the observable state space of each agent, $\mathcal{A}$ represents the action space of each agent, and $\mathcal{R}$ represents the reward function. Common MDP is mainly used to model the traditional decision-making process. However, the time between decision-making stages is not constant in the actual scene. Thus, it can be hard for the traditional MDP to model such problems. The SMDP, as an extension of the MDP, is an option-based decision-making process that can model this kind of control problem with variable decision-making times.

An SMDP is mainly composed of a quintuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{R} \rangle$, where $\mathcal{O}$ represents the set of options, and other elements in the tuple have the same meaning as in the MDP. The ption can be represented by a triplet $\langle I_o, \pi_o, \beta_o \rangle$, where $I_o \in \mathcal{S}$ is the initial state set of the option, π_o represents the strategy adopted within the option, and β_o represents the termination condition of the option. In other words, the option $\langle I_o, \pi_o, \beta_o \rangle$ is available only when the current state $s_t \in I_o$. If the option is chosen, the agent has to choose the next behavior according to the strategy π, and β_o is responsible for judging whether it reaches the termination state. If so, the agent chooses the next option to enter. The MDP can be converted into an SMDP by defining the set $\mathcal{O}$ of options on the basis of MDP. The original MDP is more like a special case of the SMDP, in which each action in the MDP is an option that only lasts for a moment.

The upper-level strategy in the SMDP $\mu : \mathcal{S} \times \mathcal{O} \to [0, 1]$ applies to the reinforcement learning of the selection of an option; that is, to choose an option from $\mathcal{O}$ according to the current state, and then proceed to the next round of selection after the option reaches the terminate state. As for the strategy π in each option, it can include some pre-defined rules, or a mapping function from a state to an action space learned through a reinforcement learning algorithm.

4.2. Proposed Model

We consider that the UAV can return to the control station for charging when the battery is about to run out, and then continue its data collection task. This paper uses the SMDP to model the UAV-assisted data collection process, and regards the UAV responsible for data collection as an agent, which is designed as follows.

- State space model $\mathcal{S}$: The system state space includes the state of the UAV and the AoI of the sensor equipment. At each time t, the system state is expressed as

$s_t = \{p_u(t), e(t), \triangle(t)\}$. Among them, $p_u(t)$, $e(t)$ are the own state of the UAV, representing the current position and remaining energy of the UAV, respectively, and $\triangle(t) = \{\triangle_1(t), \ldots, \triangle_n(t)\}$ represents the current AoI of each sensing device.

- Work space model $\mathcal{A}$: In this paper, the no-fly areas due to terrain and other reasons are not considered, and the UAV can reach the target area by flying in a straight line. Therefore, the action space includes three categories: a_f, a_c, and a_r, where $a_f = \{a_{f,0}, a_{f,1}, ..., a_{f,n}, ..., a_{f,N}\}, n \in \mathcal{N}$ represents the flight action of the UAV, and $a_{f,n}$ is the action of the UAV flying straight from the current position to the n-th sensing device, in which $a_{f,0}$ represents the UAV flying in a straight line above the control station. a_c indicates that the UAV performs a data collection action, and a_r indicates that the UAV performs a charging action.
- Option set $\mathcal{O}$: The set of options to be executed by the UAV agent in this chapter includes o_c, o_r, where $o_c = \{o_{c,1}, \ldots, o_{c,n}, \ldots, o_{c,N}\}, n \in \mathcal{N}$, and $o_{c,n}$ represents the data of the n-th sensor device for the collection tasks. The UAV first flies in a straight line from the current position to the top of the n-th sensing device, executing the action $a_{f,n}$, and then executes the data collection action a_c. The completion of data collection is the suspension state. o_r represents the option of returning home for charging. The strategy of this option is that the UAV first flies from the current position to the control station executing the action $a_{f,0}$, and then executes the charging action a_r. The suspension state means that the power of the UAV reaches E_{max}. In this SMDP, the selectable option set of the agent in each state is the entire option set $I_o = \mathcal{S}$.
- Reward function $\mathcal{R}$: In the SMDP constructed in this paper, the agent can only get rewards after executing an option, and all options use the same reward function. The instantaneous reward function of the i-th decision can be expressed as

$$r_{o,i} = \begin{cases} \frac{1}{T_i} \sum_{t=\sum_{j=1}^{i-1} T^j}^{\sum_{j=1}^{i} T_j} Q(t) + \beta_e \frac{e(\sum_{j=1}^{i} T_j)}{E_{max}} & \text{for} \quad e(\sum_{j=1}^{i} T_j) \geq e_{min}, \\ p & \text{for} \quad e(\sum_{j=1}^{i} T_j) < e_{min}. \end{cases} \tag{12}$$

It can be seen from the above formula (12) that there are two main situations for instantaneous rewards. e_{min} represents the minimum energy consumption that can guarantee the return of the UAV. After an option is executed, it will bring a penalty value p if the energy consumption of the UAV is less than the threshold. Otherwise, the instantaneous reward is mainly composed of two parts: the first item represents the average QoS during the execution of the option, and the second item represents the reward brought by the current remaining energy consumption. β_e represents the weight of energy consumption rewards, which is used to balance the proportion of the energy consumption rewards with the overall rewards.

4.3. Proposed H-DQN Algorithm

We use the SMDP to model the data collection and flight process of the UAV. In fact, the entire decision-making process is layered. The upper-level strategy u is used to select an option from the option set, and the strategy π_o in the option set is used to perform a series of bottom-level actions to complete the goal of the option. In the more typical deep reinforcement learning methods used to solve the SMDP problems, such as the hierarchical DQN (H-DQN) algorithm, both u and π_o need to be trained. For example, in a scene where there is a no-fly zone, the high-level strategy u is used to learn which option should be executed in the current state; the low-level strategy π_o learns the flight rules in the target area, bypassing the no-fly zone to reach the target point in the option. However, in the modeling of the SMDP in the above section, in order to simplify the scene, it is assumed that there is no no-fly zone, so the low-level strategy π_o can reach the target point in a straight flight. This is a pre-defined strategy that does not need to be learned. This chapter uses the H-DQN architecture to solve the SMDP problem.

The H-DQN algorithm needs to first initialize the Q network parameters, experience the playback pool, and initialize the environment state and decision index in each training

round. For each decision cycle, the agent selects an option o_i based on the current system state and a Q network based on the greedy strategy ε. It randomly selects an option from the option space with the probability of ε, and selects an option $o_i = \arg\max_o Q(s_i, o \mid \theta)$ according to the Q network with the probability of $1 - \varepsilon$. Then, it executes the strategy of the option. After the execution is complete, the system reaches the new state s_{i+1} and gets the reward $r_{o,i}$, saving the four-tuple $\langle s_i, o_i, s_{i+1}, r_{o,i} \rangle$ into the experience replay pool. This algorithm does not use a fixed greedy coefficient, but allows itself to be continuously updated during the training process. The update formula can be given by

$$\varepsilon = e^{\omega_d n_{epi}}, \tag{13}$$

where n_{epi} represents the current episode round number, and ω_d is the decay weight. ε will continue to decrease during the training process, together with the probability of exploration. When ω_d is smaller, ε decreases faster. After updating the greedy coefficient, the time stamp is updated according to (3) and (4). After that, K quadruple samples are randomly sampled from the empirical playback pool and the target value is calculated. The calculation formula for the target value can be expressed as

$$y_k = \begin{cases} r_{o,k} & \text{when} \quad s'_k \text{ is the terminal state,} \\ r_{o,k} + \gamma \max\limits_{a'} Q(s'_k, o' \mid \theta') & \text{when} \quad s'_k \text{ is not the terminal state.} \end{cases} \tag{14}$$

Then, $\sum_{k=1}^{K}(y_k - Q(s_k, o_k \mid \theta))$ is used as the objective loss function to update the parameters of the Q network. The H-DQN algorithm flow is shown as Algorithm 1. Every C updates cycles, and the parameters of the Q network are assigned to the target network for updating. When the remaining energy of the UAV is less than 0 or the mission execution time is reached, the current training round ends and the next round of training is entered.

Algorithm 1 The H-DQN algorithm

Input: initial Q network parameter θ and target network parameter θ'
Output: trained network parameters

initialize the experience playback pool $\mathcal{D}$ and the greedy coefficient $\varepsilon = 1$
for $episode \in \{1, 2, 3...\}$ **do**
  initialize the system state $t = 0$
  for $i \in \{1, 2, ..., I\}$ **do**
    the UAV agent gets the current system state s_i
    choose an option based on ε-greedy strategy o_i
    execute the middle strategy π_o of option
    get the new system state s_{i+1} and reward $r_{o,i}$
    save $\langle s_i, o_i, s_{i+1}, r_{o,i} \rangle$ into $\mathcal{D}$
    update the greedy coefficient ε according to (13), update $t \leftarrow t + T_i$
    randomly sample K samples from $\mathcal{D}$: $\langle s_k, o_k, s'_k, r_{o,k} \rangle, \forall k \in \{1, \ldots, K\}$
    calculate the target value y_k of each sample according to (14)
    take $\sum_{k=1}^{K}(y_k - Q(s_k, o_k \mid \theta))$ as the objective function
    use the gradient descent method to update θ
    if $i\%C = 0$ **then**
      $\theta' \leftarrow \theta$
    else if $e(t) < 0 \| t \geq T$ **then**
      $episode \leftarrow episode+1$,end this round of training
    else
      continue training

5. Performance Evaluation

In this section, the performance of the proposed H-DQN method is numerically evaluated. The simulation program runs on the Ubuntu16.04 operating system. The deep

reinforcement learning environment is realized based on the Gym toolkit, andit implements the H-DQN algorithm code based on PyTorch. We consider using a 500 m × 500 m area as a simulation scenario. Due to natural disasters or other reasons, UAVs are needed to assist in real-time data collection tasks in this area, and a certain number of sensor devices are distributed. The data convergence center is located at the edge of the area [0, 0], and the UAVs take off or land from here. During the execution of the data collection task, the UAV can return to the data convergence center for charging. In the simulation environment, the UAV flies at a fixed speed of 20 m/s and the maximum energy it can carry is 2 $\times 10^4$ J. The experimental analysis is carried out according to the percentage of residual energy. More detailed environmental parameters and UAV parameters are shown in Table 1.

Table 1. The environmental parameters and the UAV parameters.

Symbol	Parameter Description	Value
β_0	Channel gain when the distance is 1 m	−50 dB
B	Channel bandwidth	1 MHz
σ^2	Gaussian white noise power	−100 dBm
v^u	The flying speed of UAV	20 m/s
a	Utility function parameter	0.8
w	Utility function parameter	1
p_n	Transmitting power of sensor equipment	0.5 W
P_0	Blade contour energy consumption in hovering state	99.66 W
P_1	Induction energy consumption in hovering state	120.16 W
U^{tip}	Tip speed	120 m/s
δ	The time length of each time slot	1 s
v_0	Average rotor-induced speed in hovering state	0.002 m/s
d_0	Airframe drag ratio	0.48
s_0	Rotor stability	0.0001
A_r	Rotor area	0.5 s^2
E_{max}	The maximum energy that a drone can carry	2 $\times 10^4$ J

In our H-DQN method, both the Q network and the target Q network use four-layer artificial neural network architecture. The activation function is unified using the ReLu function, the output layer is added with a layer of the Softmax function to obtain the probability distribution of the output action, and the learning rate is set to 0.001. Other hyperparameters related to the H-DQN training are shown in Table 2.

Table 2. The hyperparameters of the H-DQN.

Parameter	Value
Number of training round episodes	1000
Target network update frequency C	100
Size of reply buffer $\mathcal{B}$	5000
Mini-batch size K	128
Discount factor γ	0.9
The number of neurons in the first hidden layer	128
The number of neurons in the second hidden layer	64

5.1. Training Results

First, we verify the convergence effect of the algorithm when the learning rate is 0.001, and we compare it with the traditional DQN algorithm based on rasterization processing and the Markov decision process modeling. As shown in Figure 3, it can be seen that the layered deep reinforcement learning algorithm H-DQN proposed in this paper converges around the 100th training round, and the reward remains between 140–150. In comparison, the traditional DQN algorithm has an upward trend, but the convergence process is much slower than the H-DQN method proposed in this paper. This is due to the large action state space of the traditional DQN algorithm, while the H-DQN method is modeled through

the SMDP and the option, which greatly reduces the action state space, making it easier to learn the optimal strategy. In addition, in the last 200 training rounds, the DQN method has a certain convergence trend, but the reward is still lower than the H-DQN method. This is because the rasterized flight mode of the DQN method causes more energy loss.

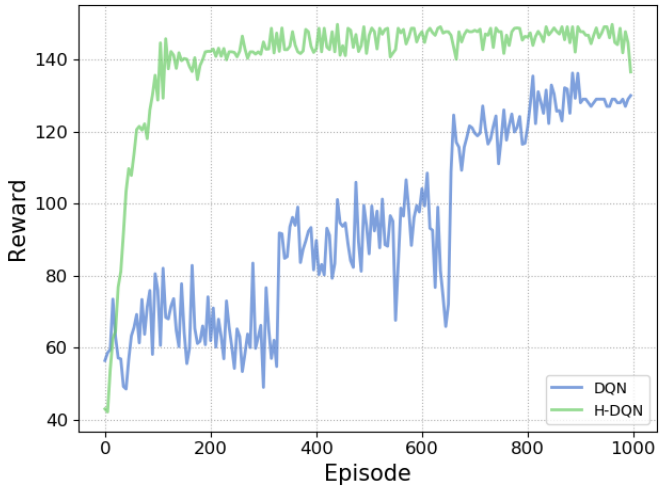

Figure 3. Convergence of the H-DQN method at different learning rates.

The energy consumption weight in the reward function will affect the performance of the algorithm. Figure 4 represents the remaining energy change curve of the H-DQN algorithm when the energy consumption weight $\beta_e = 1, 1.5, 2$. The horizontal axis represents the decision sequence, and the vertical axis represents the current remaining energy consumption as a percentage of the initial energy consumption. It can be seen that when $\beta_e = 1$, the percentage of the lowest remaining energy is 2–10%,while 2% of the remaining energy is a relatively dangerous situation, which is close to energy exhaustion, and the safety factor is low, which may lead to the UAV not being able to complete the return flight, making it difficult to deal with emergencies. With the increase in β_e, it can be seen that the charging frequency of the UAV also increases, and the remaining energy can also be maintained at a safer level. When $\beta_e = 1.5$, the remaining energy of the UAV is always maintained above 18%. When $\beta_e = 2$, the remaining energy of the UAV is basically maintained at more than 40%, but frequent charging may cause more QoS losses. As shown in Figure 5, when $\beta_e = 1$, the average QoS of the system is maintained at around 0.25, which is significantly higher than the QoS when $\beta_e = 1.5$ and $\beta_e = 2$. Therefore, users can set β_e according to the application requirements to achieve a tradeoff between the QoS and the safe power state.

5.2. Comparing with Baselines

In this section, we test the effect of the H-DQN algorithm and other benchmark algorithms. The evaluation metric is the average QoS. As shown in (7), the QoS is a utility function about the AoI. The benchmark algorithms include the DQN algorithm and the only AoI selection method. The DQN algorithm uses the traditional MDP to model the path-planning problem, in which the target area is divided into 20 × 20 square sub-areas of equal area, and the center point of each sub-area is specified as the waypoint where the UAV can hover. The action space of the UAV is to move to an adjacent sub-area, and perform data collection tasks and charging tasks in this area. The instantaneous reward of the UAV is set to the AoI QoS only, and the energy reward of the time slot. In the AoI only method, each time the UAV selects the sensor device with the largest AoI in the current system for data collection, it then returns for recharging when the remaining power is less than a certain threshold.

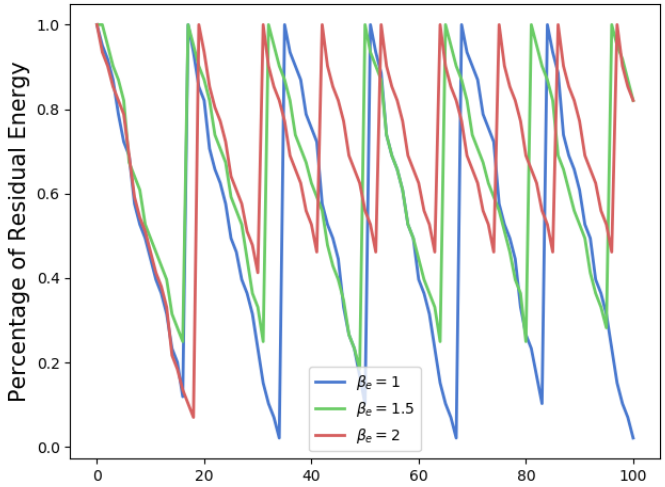

Figure 4. Change curve of residual energy consumption of the UAV.

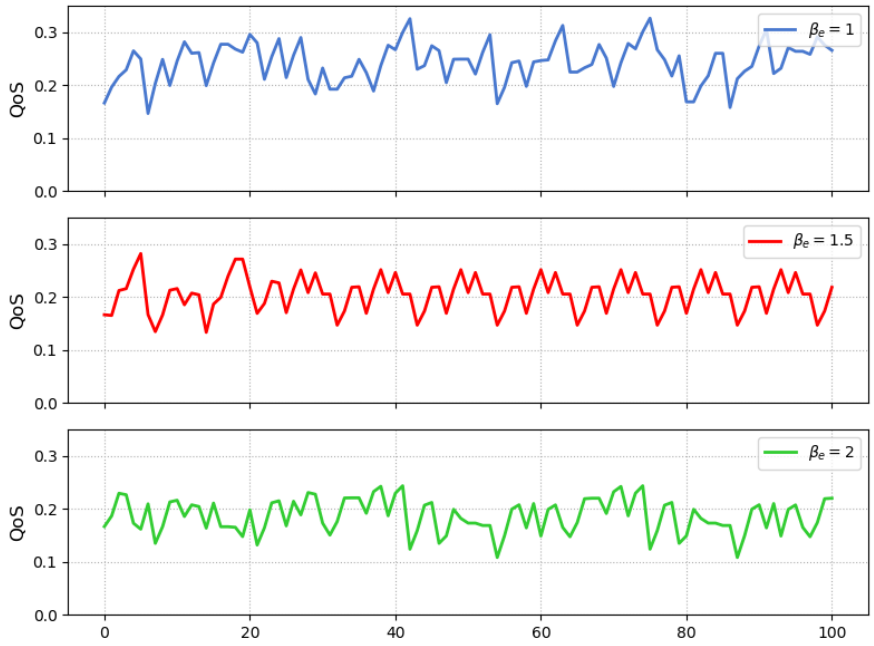

Figure 5. QoS change curve of the UAV-assisted data collection system.

Figure 6 describes the performance of the H-DQN and other benchmark algorithms with different numbers of sensing devices. It can be seen that when the number of ground sensors is 10, 20, and 30, the performance of the H-DQN algorithm is better than the other two benchmark algorithms. First of all, the selection algorithm based on the AoI is a greedy strategy, and the effect in long-term tasks is not as good as reinforcement learning methods, while reinforcement learning methods focus on long-term cumulative benefits. In addition, the performance of the H-DQN algorithm, based on the SMDP modeling, is also better than the traditional DQN method which is based on the MDP modeling. When the number of ground sensors is 10, 20, and 30, the performance of the H-DQN method is improved by 18.36%, 37%, and 21%, respectively, compared with the DQN method. This is because the H-DQN method, based on the SMDP modeling, uses the concept of time abstraction, which greatly reduces the state-action space, making it easier to learn better strategies. However, as the number of sensing devices increases, the performance of the H-DQN algorithm

gradually declines. Therefore, when the number of sensing devices is large, the use of the multi-UAV collaboration for data collection should be considered.

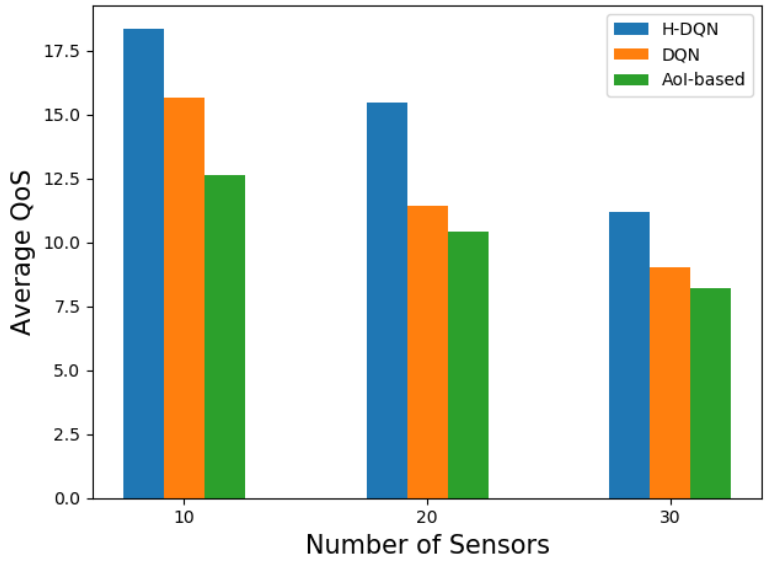

Figure 6. Comparison of the H-DQN and benchmark algorithms.

6. Conclusions

In this paper, we have studied the path-planning problem in UAV-assisted data collection. In order to enable UAVs to perform long-term data collection tasks in dynamic scenarios with a constantly changing AoI and their own power levels, we propose a method based on the QoS maximization problem of the AoI, and we use a semi-Markov decision process (SMDP) to describe this problem. In addition, we also propose a path-planning algorithm based on hierarchical DQN to learn the optimal strategy, and conduct simulation experiments to verify the convergence of the proposed algorithm. We test the effect of the energy consumption weight β_e on the remaining power and the QoS influence. Experiments show that the tradeoff between the system QoS and the safe power state can be achieved by adjusting the parameters β_e. At the same time, it is compared with two benchmark algorithms, i.e., the DQN and the AoI only algorithm. The results show that this algorithm is better than the other two benchmark algorithms. In the H-DQN method in this paper, the bottom-level strategy option uses the established strategy. In future work, further improvements can be made, such as using the DQN for high-level strategies and the DDPG for low-level strategies.

Author Contributions: Conceptualization, Z.Q. and X.Z. (Xuan Zhang); methodology, Z.Q., X.Z. (Xuan Zhang) and B.L.; validation, Z.Q., X.Z. (Xuan Zhang) and X.Z. (Xinwei Zhang); formal analysis, B.L.; investigation, Z.L. and L.G.; resources, B.L.; data curation, Z.L.; writing—original draft preparation, Z.Q.; writing—review and editing, X.Z. (Xuan Zhang); visualization, L.G.; project administration, B.L. and L.G. All authors have read and agreed to the published version of the manuscript.

Funding: The work was supported by "National Natural Science Foundation of China" with No. 61902052, "Science and Technology MajorIndustrial Project of Liaoning Province" with No. 2020JH1/10100013, "Dalian Science and Technology Innovation Fund" with No. 2019J11CY004 and 2020JJ26GX037, and "the Fundamental Research Funds for the Central Universities" with No. DUT20ZD210 and DUT20TD107.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The relevant data of simulation experiment can be found in https://github.com/wilna-lab/H-DQN.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yong, Z.; Rui, Z.; Teng, J.L. Wireless Communications with Unmanned Aerial Vehicles: Opportunities and Challenges. *IEEE Commun. Mag.* **2016**, *54*, 36–42.
2. Mozaffari, M.; Saad, W.; Bennis, M.; Nam, Y.H.; Debbah, M. Fundamental Tradeoffs in Communication and Trajectory Design for UAV-Enabled Wireless Network. *IEEE Wirel. Commun.* **2019**, *26*, 36–44.
3. Cileo, D.G.; Sharma, N.; Magarini, M. Coverage, Capacity and Interference Analysis for An Aerial Base Station in Different Environments. In Proceedings of the 2017 International Symposium on Wireless Communication Systems (ISWCS), Bologna, Italy, 28–31 August 2017; pp. 281–286.
4. Dm, A.; En, B. A Survey on Cellular-Connected UAVs: Design Challenges, Enabling 5G/B5G Innovations, and Experimental Advancements. *Comput. Netw.* **2020**, *182*, 107451.
5. Zhao, N.; Lu, W.; Sheng, M.; Chen, Y.; Tang, J.; Yu, F.R.; Wong, K.K. UAV-Assisted Emergency Networks in Disasters. *IEEE Wirel. Commun.* **2019**, *26*, 45–51. [CrossRef]
6. Wu, Q.; Liu, L.; Zhang, R. A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems. *IEEE Wirel. Commun.* **2019**, *26*, 36–44. [CrossRef]
7. Yang, D.; Wu, Q.; Zeng, Y.; Zhang, R. Energy Trade-off in Ground-to-UAV Communication via Trajectory Design. *IEEE Trans. Veh. Technol.* **2017**, *67*, 6721–6726. [CrossRef]
8. Chiaraviglio, L.; Amorosi, L.; Malandrino, F.; Chiasserini, C.F.; Dell'Olmo, P.; Casetti, C. Optimal Throughput Management in UAV-based Networks during Disasters. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; pp. 307–312.
9. Sharafeddine, S.; Islambouli, R. On-Demand Deployment of Multiple Aerial Base Stations for Traffic Offloading and Network Recovery. *Comput. Netw.* **2019**, *156*, 52–61. [CrossRef]
10. Abdulla, A.E.; Fadlullah, Z.M.; Nishiyama, H.; Kato, N.; Ono, F.; Miura, R. An Optimal Data Collection Technique for Tmproved Utility in UAS-Aided Networks. In Proceedings of the IEEE INFOCOM 2014—IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April–2 May 2014; pp. 736–744.
11. Zhan, C.; Zeng, Y.; Zhang, R. Energy-Efficient Data Collection in UAV Enabled Wireless Sensor Network. *IEEE Wirel. Commun. Lett.* **2017**, *7*, 328–331. [CrossRef]
12. Wang, Z.; Zhang, G.; Wang, Q.; Wang, K.; Yang, K. Completion Time Minimization in Wireless-Powered UAV-Assisted Data Collection System. *IEEE Commun. Lett.* **2021**, *25*, 1954–1958. [CrossRef]
13. Samir, M.; Sharafeddine, S.; Assi, C.M.; Nguyen, T.M.; Ghrayeb, A. UAV Trajectory Planning for Data Collection from Time-Constrained IoT Devices. *IEEE Trans. Wirel. Commun.* **2019**, *19*, 34–46. [CrossRef]
14. Cheng, F.; Zhang, S.; Li, Z.; Chen, Y.; Zhao, N.; Yu, F.R.; Leung, V.C. UAV Trajectory Optimization for Data Offloading at the Edge of Multiple Cells. *IEEE Trans. Veh. Technol.* **2018**, *67*, 6732–6736. [CrossRef]
15. Liu, C.H.; Chen, Z.; Tang, J.; Xu, J.; Piao, C. Energy-Efficient UAV Control for Effective and Fair Communication Coverage: A Deep Reinforcement Learning Approach. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 2059–2070. [CrossRef]
16. Li, K.; Ni, W.; Tovar, E.; Guizani, M. Joint Flight Cruise Control and Data Collection in UAV-Aided Internet of Things: An Onboard Deep Reinforcement Learning Approach. *IEEE Internet Things J.* **2021**, *8*, 9787–9799. [CrossRef]
17. Liu, Q.; Shi, L.; Sun, L.; Li, J.; Ding, M.; Shu, F. Path Planning for UAV-Mounted Mobile Edge Computing With Deep Reinforcement Learning. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5723–5728. [CrossRef]
18. Gong, J.; Chang, T.H.; Shen, C.; Chen, X. Flight Time Minimization of UAV for Data Collection Over Wireless Sensor Networks. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 1942–1954. [CrossRef]
19. Kaul, S.; Yates, R.; Gruteser, M. Real-Time Status: How Often Should One Update? In Proceedings of the 2012 Proceedings IEEE INFOCOM, Orlando, FL, USA, 25–30 March 2012; pp. 2731–2735.
20. Arulkumaran, K.; Deisenroth, M.P.; Brundage, M.; Bharath, A.A. Deep Reinforcement Learning: A Brief Survey. *IEEE Signal Process. Mag.* **2017**, *34*, 26–38. [CrossRef]
21. Qian, Y.; Wu, J.; Wang, R.; Zhu, F.; Zhang, W. Survey on Reinforcement Learning Applications in Communication Networks. *J. Commun. Inf. Netw.* **2019**, *4*, 30–39.
22. Abd-Elmagid, M.A.; Ferdowsi, A.; Dhillon, H.S.; Saad, W. Deep Reinforcement Learning for Minimizing Age-of-Information in UAV-assisted Networks. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6.
23. Zhou, C.; He, H.; Yang, P.; Lyu, F.; Wu, W.; Cheng, N.; Shen, X. Deep RL-based Trajectory Planning for AoI Minimization in UAV-assisted IoT. In Proceedings of the 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xi'an, China, 23–25 October 2019; pp. 1–6.
24. Sun, Y.; Kadota, I.; Talak, R.; Modiano, E. *Age of Information: A New Metric for Information Freshness*; Morgan & Claypool: San Rafael, CA, USA, 2019.
25. Samir, M.; Assi, C.; Sharafeddine, S.; Ebrahimi, D.; Ghrayeb, A. Age of Information Aware Trajectory Planning of UAVs in Intelligent Transportation Systems: A Deep Learning Approach. *IEEE Trans. Veh. Technol.* **2020**, *69*, 12382–12395. [CrossRef]

26. Cao, A.; Shen, C.; Zong, J.; Chang, T.H. Peak Age-of-Information Minimization of UAV-Aided Relay Transmission. In Proceedings of the 2020 IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
27. Yi, M.; Wang, X.; Liu, J.; Zhang, Y.; Bai, B. Deep Reinforcement Learning for Fresh Data Collection in UAV-assisted IoT Networks. In Proceedings of the 2019 IEEE International Conference on Communications Workshops (ICC Workshops), Shanghai, China, 20–24 May 2019; pp. 716–721.
28. Sutton, R.S.; Barto, A.G. *Reinforcement Learning*; A Bradford Book; MIT Press: Cambridge, MA, USA,1998; Volume 15, pp. 665–685.
29. Ivanov, S.; D'Yakonov, A. Modern Deep Reinforcement Learning Algorithms. *arXiv* **2019**, arXiv:1906.10025v1.
30. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**, arXiv:1312.5602v1.
31. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-Level Control Through Deep Reinforcement Learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
32. Wang, Q.; Zhang, W.; Liu, Y.; Liu, Y. Multi-UAV Dynamic Wireless Networking with Deep Reinforcement Learning. *IEEE Commun. Lett.* **2019**, *23*, 2243–2246. [CrossRef]
33. Yi, M.; Zhang, Y.; Wang, X.; Xu, C.; Ma, X. Deep Reinforcement Learning for User Association in Heterogeneous Networks with Dual Connectivity. In Proceedings of the 2021 IEEE Wireless Communications and Networking Conference (WCNC), Nanjing, China, 29 March–1 April 2021; pp. 1–5.
34. Mao, C.; Liu, J.; Xie, L. Multi-UAV Aided Data Collection for Age Minimization in Wireless Sensor Networks. In Proceedings of the 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 21–23 October 2020; pp. 80–85.
35. Abedin, S.F.; Munir, M.S.; Tran, N.H.; Han, Z.; Hong, C.S. Data Freshness and Energy-Efficient UAV Navigation Optimization: A Deep Reinforcement Learning Approach. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 5994–6006. [CrossRef]