processes

MDPI

Article

Optimizing Task Completion Time in Disaster-Affected Regions with the WMDDPG-GSA Algorithm for UAV-Assisted MEC Systems

Tianhao Ma, Yulu Yang, Han Xu and Tiecheng Song *

School of Information Science and Engineering, Southeast University, Nanjing 210096, China; 220210899@seu.edu.cn (T.M.); 220210753@seu.edu.cn (Y.Y.); han_xu@seu.edu.cn (H.X.)
* Correspondence: songtc@seu.edu.cn

Abstract: In this paper, we investigate a UAV-assisted Mobile Edge Computing (MEC) system that is deployed with multiple UAVs to provide timely data processing services to disaster-stricken areas. In our model, since base stations are unavailable in disaster-affected areas, we solely employ UAVs as MEC servers, as well as enable real-time data transmission during UAV flights by estimating and compensating for the Doppler Frequency Shift (DFS). Subsequently, an optimization problem is formulated to jointly optimize the trajectories and offloading strategies of multiple UAVs to minimize the task completion time. We enhance the performance of the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm by using a weighted strategy algorithm, and we thus propose the Weighted-Strategy-Based Multi-Agent Deep Deterministic Policy Gradient (WMDDPG) algorithm for optimizing UAV trajectories. We employ the Greedy-Based Simulated Annealing (GSA) algorithm to overcome the limitations of the greedy algorithm and to obtain the best offloading strategy. The results demonstrate the effectiveness of the proposed WMDDPG-GSA algorithm, as it outperforms benchmark algorithms.

Keywords: UAV; Mobile Edge Computing (MEC); Multi-Agent Deep Reinforcement Learning; Doppler Frequency Shift (DFS)

Citation: Ma, T.; Yang, Y.; Xu, H.; Song, T. Optimizing Task Completion Time in Disaster-Affected Regions with the WMDDPG-GSA Algorithm for UAV-Assisted MEC Systems. *Processes* **2023**, *11*, 3000. https://doi.org/10.3390/pr11103000

Academic Editor: Iqbal M. Mujtaba

Received: 24 September 2023
Revised: 13 October 2023
Accepted: 13 October 2023
Published: 18 October 2023

1. Introduction

Edge computing has attracted extensive attention for its ability to offer faster response times and a higher quality of service to User Equipment (UE) in the context of the Internet of Things (IoT) [1]. However, in areas affected by natural disasters such as earthquakes, the limited power supply of traditional edge computing nodes hampers their ability to provide data processing services. This limitation obstructs timely feedback on disaster-related data and hinders the progress of relief efforts. In conventional edge computing systems, unmanned aerial vehicles (UAVs) serve as communication relays to facilitate edge computing by establishing a line-of-sight (LoS) channel with terrestrial UE [2]. In recent years, the enhanced mobility and computing capabilities of UAVs have sparked interest in utilizing them as mobile edge servers to deliver reliable data processing services for UE in disaster-stricken areas [3]. This development has given rise to UAV-assisted Mobile Edge Computing (MEC) systems [4].

Ensuring the timely processing and feedback of data during disaster rescue operations is crucial in mitigating the impact of secondary disasters on lives and property. Therefore, timeliness becomes a paramount consideration in UAV-assisted MEC systems [5]. Consequently, this paper adopts timeliness as the evaluation criterion for UAV-assisted MEC systems in disaster-affected areas or in other energy-constrained regions. The following section introduces UAV-assisted MEC systems, explores the impact of the DFS on these systems, discusses the current research on UAV trajectory design and offloading strategy, and then finally introduces the main contributions and structure of this paper.

1.1. The UAV-Assisted MEC System

In previous studies, due to significant signal obstructions caused by urban infrastructure, some UAV-assisted MEC systems employed UAVs as communication relays [6,7]. The utilization of UAVs as communication relays and MEC servers is a common approach in UAV-assisted MEC systems [8,9]. Limited studies have explored exclusively utilizing UAVs as MEC servers [10].

However, these studies do not consider the situation in which ground base stations in disaster-stricken areas are unable to provide services due to power limitations. Therefore, in order to better reflect the actual situation, this paper only considers UAVs as MEC servers to assist the MEC system. Furthermore, to fully exploit the computing power and data transmission potential of UAVs during flight, we combine data transmission with UAVs movement and employ the Orthogonal Frequency Division Multiplexing (OFDM) system to effectively meet the timeliness requirements.

1.2. The Impact of the DFS on the UAV-Assisted MEC System

With the application of 5G and B5G (beyond 5G) technologies in a MEC system, along with the Doppler Frequency Shift (DFS) caused by the high mobility of UAVs, the subcarriers of the OFDM communication link are no longer orthogonal; as a result, the receiver cannot correctly demodulate the signal [11–13]. Therefore, in order to achieve data transmission during UAV flights, it is necessary to consider DFS estimation and compensation in the transmission process.

The research on UAV-assisted MEC systems has produced a data-assisted DFS estimation and compensation method, whereby the DFS estimation process has been optimized by utilizing historical estimation results to achieve rapid and accurate DFS compensation [14]. However, this research is limited to the communication link between UAVs and UE without encompassing specific MEC scenarios. In the papers of [15,16], a more practical G2A channel model was used, and it was assumed that the DFS caused by UAVs was well estimated. However, the above studies did not use actual algorithms to estimate the DFS.

1.3. UAVs Trajectory Design

Currently, UAV-assisted MEC systems can be divided into two categories: single-UAV systems and multi-UAV systems. In single-UAV systems, due to the highly non-convex nature of UAV trajectory design, conventional research approaches often employ strategies based on continuous convex approximation [17,18]. With the advancement of deep reinforcement learning (DRL), the DRL techniques based on discrete or continuous action spaces have been employed in UAV-assisted edge systems. The commonly used algorithms include DDPG [19], Q-learning [20], etc.

The Multi-UAV-assisted MEC systems offer numerous advantages compared to single-UAV systems. For example, the collaboration among UAVs can effectively cater to a larger number of UE simultaneously, thereby further reducing the overall service duration. In earlier studies, a meta-heuristic ant colony optimization algorithm was proposed to optimize the trajectory of UAVs [21]. Since traditional DRL algorithms such as DDPG can only be used for single-agent tasks, they are not suitable for multi-UAV systems. However, the integration of multi-agent learning and deep reinforcement learning has led to the development of algorithms such as MADDPG [22] and COMA [23], which have simplified the solution for multi-UAV systems. As a result, multi-agent reinforcement learning (MADRL) has gained popularity in the field of multi-UAV systems. For example, both the papers of [24,25] utilized the MADDPG algorithm to optimize UAV trajectories.

1.4. Offloading Strategy

After determining the flight trajectory of UAVs, different offloading strategies will affect the timeliness of the system. Most studies on UAV-assisted MEC systems do not have specific offloading strategy algorithms [24,26]. If we consider UE as tourists and different UAVs as different means of transportation, this offloading strategy can be seen as a multi-

criteria itinerary planning strategy. In [27], a state-transition-simulated annealing algorithm was proposed to find the shortest path for multi-criteria journey planning. Considering the different interests tourists have in different attractions, a greedy algorithm was used to minimize the total distance of the journey [28]. The above algorithms can be used as offloading strategies in this study, but the simulated annealing algorithm may have the problem of not finding the optimal solution, while the greedy algorithm may fall into the issue of local optima.

1.5. Contributions and Structure

This paper considers a UAV-assisted MEC system for disaster scenarios, where multiple UAVs are deployed as MEC servers. The main contributions of this paper are as follows:

1. This paper presents a UAV-assisted MEC system that models the UAV trajectory as a continuous variable. This approach offers a more accurate representation of the UAV's behavior and performance during movement.
2. In our model, we enable data transmission during UAV movement by considering the impact of the DFS on data transmission. The accurate estimation of the DFS is employed to enhance the precision of data transmission during UAV movement.
3. To tackle this highly non-convex problem, we decompose it into two sub-problems in this study. The first sub-problem addresses UAV trajectory planning, for which we propose a Weighted-Strategy-Based Multi-Agent Deep Deterministic Policy Gradient (WMDDPG) algorithm to determine the optimal trajectory. The second sub-problem deals with optimizing the offloading strategy. To overcome the limitations of a greedy algorithm, we introduce an enhanced approach that combines both greedy and simulated annealing (GSA) algorithms.

The remaining sections of this paper are structured as follows: Section 2 presents the system model and problem formulation, which is followed by our proposed WMDDPG-GSA algorithm in Section 3 to address the optimization problem. Subsequently, we present the simulation results in Section 4, and then conclude with Section 5.

2. System Model

As shown in Figure 1, this paper presents a UAV-assisted MEC system that consists of multiple UAVs collaborating to provide services to a UE group. The UAVs primarily receive data from the UE and process it, which is denoted by $\mathcal{M} = \{1, 2, \ldots, M\}$. It is assumed in this paper that there are N randomly distributed UE in the system, which is represented by $\mathcal{N} = \{1, 2, \ldots, N\}$.

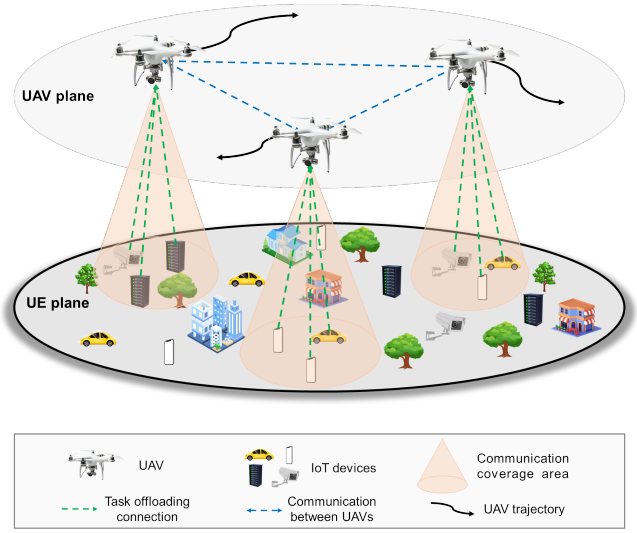

Figure 1. System model.

2.1. Motion Model

In order to minimize the service time, this paper discretizes the total service time into K_{max} time slots, each of length Δ, which is represented by $\mathcal{K} = \{1, 2, \ldots, K_{max}\}$. The UAVs move to the next destination in each time slot and can receive and send data during the flight. The position of the l-th UAV after the end of the k-th time slot (when $k = 0$, the position of the UAV is the starting position) is denoted as follows:

$$\boldsymbol{q}_l(k) = [x_l(k),\ y_l(k)]^T, \forall l \in \mathcal{M}, \forall k \in \mathcal{K}. \tag{1}$$

The location of the i-th UE is the following:

$$\boldsymbol{w}_i = [w_i(x),\ w_i(y)]^T. \tag{2}$$

As shown in Figure 2, the l-th UAV flies at a velocity $v_l(k)$ in each time slot k. This velocity cannot exceed the maximum flight speed V_{max}. The flight direction is represented by the unit vector $\boldsymbol{\xi}_l(k)$.

$$\boldsymbol{q}_l(k) = \boldsymbol{q}_l(k-1) + v_l(k)\boldsymbol{\xi}_l(k), \forall l \in \mathcal{M}, \forall k \in \mathcal{K}, \tag{3}$$

$$\boldsymbol{\xi}_l(k) = [\cos(\varphi_l(k)), \sin(\varphi_l(k))], \forall l \in \mathcal{M}, \forall k \in \mathcal{K}. \tag{4}$$

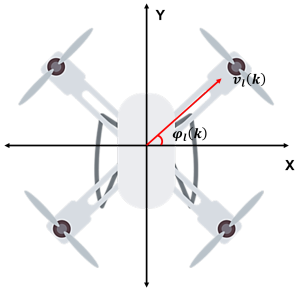

Figure 2. Schematic diagram of UAV flight.

The UAV flight area is defined as a circular area with radius R, which has the following:

$$x_l(k)^2 + y_l(k)^2 \leq R^2, \forall l \in \mathcal{M}, \forall k \in \mathcal{K}. \tag{5}$$

In order to prevent collisions between the UAVs, the positions of the UAVs should have the following relationship:

$$\|\boldsymbol{q}_1(k) - \boldsymbol{q}_{l'}(k)\|_2 \geq D_{min}, \forall l \neq l' \in \mathcal{M}, \forall k \in \mathcal{K}. \tag{6}$$

According to the paper of [29], the flight power of the UAVs can be expressed as a function of velocity:

$$P^F(v) = P_0\left(1 + \frac{3v^2}{U_{tip}^2}\right) + P_i\left(\sqrt{1 + \frac{v^4}{4v_0^4}} - \frac{v^2}{2v_0^2}\right)^{\frac{1}{2}} + \frac{1}{2}d_0\rho s G v^3, \tag{7}$$

where P_0 and P_i are constants, U_{tip}, v_0, d_0, s and G are parameters associated with UAV characteristics, and ρ is a parameter related to the environment. Therefore, the energy consumption of the l-th UAV flight in time slot k is given by the following formula:

$$E_l^F(k) = P^F(v_l(k))\Delta, \forall l \in \mathcal{M}, \forall k \in \mathcal{K}. \tag{8}$$

2.2. Communication Model

2.2.1. Communication between UAVs and UE

At the beginning of each time slot, the UE choose whether to offload data to UAVs or to perform local computing. Let $a_{i,l}(k)$ denote the offloading policy of i-th UE at time slot k, where $a_{i,l}(k) = 1$ indicates that i-th UE, which is connected to l-th UAV, otherwise $a_{i,l}(k) = 0$. In this paper, we adopt the orthogonal frequency-division multiplexing (OFDM) system, which means that, due to the limited number of subcarriers, only the maximum $C_{\max}$ UE can offload tasks to the UAVs within one time slot:

$$\sum_{i=1}^{N} a_{i,l}(k) \leq C_{\mathrm{MAX}}, \forall l \in \mathcal{M}, \forall i \in \mathcal{N}, \forall k \in \mathcal{K}. \tag{9}$$

Due to the limited transmission capability of the UAVs, the UE whose distance is greater than D_{max} cannot be linked:

$$a_{i,l}(k) d_{i,l}(k) \leq D_{max}, \forall l \in \mathcal{M}, \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \tag{10}$$

where $d_{i,l}(k)$ represents the distance between the i-th UE and l-th UAV, which is defined as follows: $d_{i,l}(k) = \sqrt{\|\boldsymbol{q}_l(k) - \boldsymbol{w}_i(k)\|_2^2 + H^2}$.

The angle between the l-th UAV and the i-th UE on slot k can be expressed as follows:

$$\theta_{i,l}^1(k) = \arcsin\left(\frac{H}{\|\boldsymbol{q}_l(k) - \boldsymbol{w}_i(k)\|_2}\right). \tag{11}$$

Considering the LoS channel between the UAVs and UE, we assumed that the UAVs can obtain the positions of the UE through the radar or cameras equipped on the UAVs. According to [30], the probability of the LoS channel is as follows:

$$P_{i,l}(k) = \frac{1}{1 + Cexp\Big(-D\Big(\theta_{i,l}^1(k) - C\Big)\Big)}. \tag{12}$$

Therefore, the average channel gain can be expressed as per the following:

$$\begin{aligned} h_{i,l}(k) &= P_{i,l}(k)\chi\beta_0 d_{i,l}^{-\alpha}(k) + (1 - P_{i,l}(k))\beta_0 d_{i,l}^{-\alpha}(k) \\ &= \hat{P}_{i,l}(k)\beta_0 d_{i,l}^{-\alpha}(k). \end{aligned} \tag{13}$$

where β_0 and χ are the attenuation coefficients, and α is the path loss exponent. $\hat{P}_{i,l}(k) = P_{i,l}(k)\chi + (1 - P_{i,l}(k))$ represents the equivalent attenuation coefficient that considers the LoS channel between the l-th UAV and the i-th UE at time slot k. Hence, the transmission rate is expressed as follows:

$$\begin{aligned} r_{i,l}(k) &= B\log_2\left(1 + \frac{h_{i,l}(k)P^{UE}}{\sigma^2}\right) \\ &= B\log_2\left(1 + \frac{\beta_0 P^{UE}\hat{P}_{i,l}(k)}{\sigma^2 d_{i,l}(k)^{\alpha}}\right). \end{aligned} \tag{14}$$

B represents the bandwidth of the communication channel, P^{UE} represents the transmission power of the UE, and σ^2 represents the variance in the Gaussian noise that is added during the transmission process. For simplicity, we defined this as follows:

$$SNR = \frac{\beta_0 P^{UE}}{\sigma^2}. \tag{15}$$

As shown in Figure 3, due to the limited energy of UE, the UE can perform partial data computation locally in the initial stage. We assume that in time slot k, the i-th UE is connected to the l-th UAV, and from then on, the data of the UE will only be computed on the UAV. $T_{i,l}^{Tr}(k)$ represents the duration of transmission between the i-th UE, which is connected to the l-th UAV in the k-th time slot, and $k_{i,l}^{link}$ indicates the time slot number when the i-th UE and l-th UAV start linking. Therefore, the number of bits transmitted by the i-th UE and l-th UAV in the k-th time slot can be expressed as per the following:

$$D_{i,l}(k) = r_{i,l}(k) T_{i,l}^{T}(k), \tag{16}$$

$$\sum_{l=1}^{M} \sum_{k=0}^{K_{max}} D_{i,l}(k) = D_i - \frac{f^{UE}\left(k_{i,l}^{link} - 1\right)\Delta}{s_i}, \ T_{i,l}^{T}(k) \leq \Delta. \tag{17}$$

where D_i represents the total amount of data for the i-th UE, and s_i represents the CPU cycles required for the i-th UE to process one bit.

Compared to offloading data, the amount of data for computing results is relatively small, so the return time can be ignored. In addition, based on the above, it can be inferred that the energy consumed by the l-th UAV during the data reception process in time slot k is as follows:

$$E_l^T(\kappa) = \sum_{i=1}^{N} T_{i,l}^{T}(k) P^{tr-UAV}, \ T_{i,l}^{T}(k) \leq \Delta, \tag{18}$$

P^{tr-UAV} is the power of the UAVs when receiving data.

Note: The yellow box indicates that the ue's data is calculated locally, while the orange box indicates that it is calculated by the drone.

Figure 3. Schematic diagram of the UE data processing.

2.2.2. Communication between UAVs

We assume that the communication between the UAVs does not interfere with the communication between the UAVs and UE. As the communication between the UAVs is mainly used to transmit computational results, the communication time between the UAVs can be ignored.

2.2.3. DFS Estimation during UAV Flight

During the flight of the l-th UAV, the received data experience the DFS. Let the original transmission frequency be f, and the known speed of flight for each time slot be $v_l(k)$. We assume that the communication between the UAVs does not interfere with the communication between the UAVs and UE. As the time slots are very small, the DFS of the initial position of each time slot can be found instead of the DFS of that time slot.

In time slot k, the vector representing the projection of the connection between the l-th UAV and i-th UE in the horizontal direction is as follows:

$$\boldsymbol{QW}_{i,l}(k) = [x_l(k) - w_i(x), \ y_l(k) - w_i(y)]^T. \tag{19}$$

The angle between the flight direction of the l-th UAV and $\boldsymbol{QW}_{i,l}(k)$ is as per the following:

$$\theta_{i,l}^{2}(k) = \arccos\left(\frac{\boldsymbol{QW}_{i,l}(k) \cdot \boldsymbol{\xi}_l(k)}{|\boldsymbol{QW}_{i,l}(k)| \times |\boldsymbol{\xi}_l(k)|}\right). \tag{20}$$

The projection of the velocity of l-th UAV at time slot k on the line connecting the l-th UAV and i-th UE can be calculated as follows:

$$v_{i,l}(k) = v_l(k) \times cos\theta_{i,l}^{2}(k) \times cos\theta_{i,l}^{1}(k). \tag{21}$$

The estimated DFS received by l-th UAV from i-th UE is per the following:

$$\Delta f_{i,l} = \frac{C}{C - v_{i,l}(k)} f - f. \tag{22}$$

After obtaining the estimated DFS, a pre-compensation algorithm can be used to correct the impact of DFS on the OFDM system [15].

2.3. Computational Model

The local computing capability of the UE is denoted by f^{UE}. Therefore, the local execution time for the UE is given by the following formula:

$$T_i^L = \frac{D_i s_i}{f^{UE}}. \tag{23}$$

The computational capability of the UAV is denoted by f^{UAV}, and it is evenly allocated to the connected UE in each time slot. Thus, the computation time of the l-th UAV for the i-th UE in time slot k (which processes the data received in time slot $k-1$) is given by the following formula:

$$T_{i,l}^C(k) = \frac{D_i(k-1)s_i}{f^{UAV}} \tag{24}$$

The computation of the DFS at each time slot k is relatively simple, and it does not require the calculation of computation time and energy consumption. The energy consumption of the l-th UAV for computation is represented as follows:

$$E_l^C(k) = \sum_{i=1}^{N} \epsilon a_{i,l}(k-1) D_{i,l}(k-1) s_i {f^{UAV}}^2, \tag{25}$$

where ϵ is a constant related to the chip structure of the UAVs.

2.4. Problem Formulation

In a disaster scenario, serving all UE as quickly as possible is of paramount importance. Therefore, under the constraint of UAV energy, we jointly optimized the trajectory and offloading strategy of the UAVs to minimize the total service time.

The total energy consumption of l-th UAV can be represented as per the following:

$$\sum_{k=0}^{K_{max}} E_l(k) = \sum_{k=0}^{K_{max}} \Big(E_l^F(k) + E_l^T(k) + E_l^C(k) \Big). \tag{26}$$

Hence, the optimization problem in this paper can be formulated as follows:

$$\begin{aligned} \mathcal{P}1: & min\ K_{max}, \\ & s.t. (1) - (6), (9), (10), \\ & \sum_{k=0}^{K_{max}} E_l(k) \le E_{max}, \end{aligned} \tag{27}$$

Here, $\boldsymbol{q}_l(k)$ and $a_{i,l}(k)$ are the parameters to be optimized.

Our objective is to jointly optimize the trajectory and offloading strategy of UAVs under the energy constraint to minimize the total service time. Therefore, the appropriate scheduling strategy and offloading strategy for UAVs are the key to our algorithm. Problem $\mathcal{P}1$ is a highly non-convex problem with integer variables $a_{i,l}(k)$ and continuous variables $\boldsymbol{q}_l(k)$, thus making it extremely difficult to find the optimal solution. To address this, we propose a WMDDPG-GSA algorithm to solve the problem.

3. WMDDPG-GSA Algorithm

In this section, we divide $\mathcal{P}1$ into two sub-problems: one is finding the optimal offloading strategy when using the GSA algorithm, and the other is finding the optimal UAVs flight trajectory when using WMDDPG.

3.1. Offloading Strategy Algorithm

Given the known UAV trajectory in time slot k, the offloading strategy algorithm aims to minimize the sum of distances between all connections while ensuring the maximum number of connections between the UAVs and UE. Therefore, the problem can be formulated as follows:

$$\begin{aligned} \mathcal{P}2: \ min_{\sum_{i=1}^{N} a_{i,l}(k)}^{max} \quad & \sum_{i=1}^{N} a_{i,l}(k) d_{i,l}(k), \\ & s.t. a_{i,l}(k) = \{0,1\}, \ \forall i,l,k, \\ & \sum_{i=1}^{N} a_{i,l}(k) \leq C_{max}, \ \forall k \in \mathcal{K}, \forall l \in \mathcal{M}, \\ & a_{i,l}(k) d_{i,l}(k) \leq \ D_{max}, \ \forall i,l,k. \end{aligned} \tag{28}$$

To address the above issue, our proposed GSA algorithm consists of two steps. First, a greedy algorithm is used to find a local optimal solution. Then, a simulated annealing algorithm is employed to use the local optimal solution generated by the greedy algorithm as a reference and search for the global optimal solution of the problem. The pseudo-code for the offloading strategy is given in Table 1.

Table 1. GSA algorithm for the offloading policy.

Algorithm GSA Algorithm for the Offloading Policy.
1: Update the distance between all UE and UAVs: $d_{i,l}(k)$.
2: Initialize the sorted list of distance between all UAVs and UE: sorted_lst, and sort it in ascending order.
3: Initialize the list of unassigned UE that can be connected to UAVs: unassigned_users.
4: Initialize the list of UE connected to each UAV as the empty list: link_dict_UAV = $[] * M$, initializes the list of UE connected to UAV in slot K as the empty list already_in = []
5: **for** $l = 1, 2, \ldots, M$ **do**
6: **for** i in sorted_lst **do**
7: **if** $d_{i,l}(k) \leq \ D_{max}$ and i not in already_in and the pending data of i-th UE > 0 **then**
8: add i to the list link_dict_UAV$[l]$ and the list already_in.
9: **end if**
10: **end for**
11: **end for**
12: store link_dict_UAV as the initial solution in best_solution, calculate the distance sum of all UE connected to the UAVs as best_cost, and calculate the total number of connected UE as max_connect_ue.
13: **for** i in range(200) **do**
14: update already_in and link_dict_UAV as empty lists.
15: **for** $l = 1, 2, \ldots, M$ **do**
16: **for** _ in range(C_{max}) **do**
17: randomly select a i'-th UE from unassigned_users.
18: remove i'-th UE from unassigned_users.
19: **if** $d_{i',l}(k) \leq \ D_{max}$ and i' not in already_in **then**
20: add i' to the list link_dict_UAV$[l]$ and the list already_in.

Table 1. *Cont.*

Algorithm GSA algorithm for the offloading policy.
end if
end for
end for
end for
Calculate the distance sum of all the UE connected to the UAVs as new_cost, and calculate the number of connected UE in the link_dict_UAV as new_connect_ue.
if new_connect_ue > new_connect_ue **then**
best_solution = link_dict_UAV
end if
if new_connect_ue == new_connect_ue **then**
if new_cost < best_cost **then**
best_solution = link_dict_UAV
end if
end if

3.2. UAV Trajectory Algorithm

3.2.1. State Space $\mathcal{S}$

In our improved MADDPG algorithm, the state is divided into private observation $o^l_{pri}(k)$ and public observation $o_{pub}(k)$:

$$o^l_{pri}(k) = \Big\{ \big[\boldsymbol{d}_{l,l'}(k), \boldsymbol{d}_{l,i}(k), d_l(k), C_l(k)\big] : \forall l' \neq l \in \mathcal{M}, \forall i \in \mathcal{N}, C_l(k) \leq 3, d_l(k) = \sqrt{x_l(k)^2 + y_l(k)^2} \Big\}, \tag{29}$$

$$o_{pub}(k) = \{[\boldsymbol{D}_i(k), T(k)] : \forall i \in \mathcal{N}\}, \tag{30}$$

where $\boldsymbol{d}_{l,l'}(k)$ represents the spatial relationship with other UAVs, $\boldsymbol{d}_{l,i}(k)$ represents the spatial relationship with UE, $d_l(k)$represents the distance between l-th UAV and the origin, and $C_l(k)$ represents the number of UE connected to l-th UAV. $\boldsymbol{D}_i(k)$ represents the remaining data of the i-th UE in time slot k, and $T(k)$ represents the elapsed time.

The observable state for the l-th UAV in time slot k is denoted as follows: $\boldsymbol{s}_l(k) = [o_{pub}(k), o^l_{pri}(k)] : \forall l \in \mathcal{M}$. The observation of the entire environment is denoted by $s(k)$:

$$\boldsymbol{s}(k) = \Big\{ \Big[o_{pub}(k), o^0_{pri}(k), o^1_{pri}(k), \ldots, o^M_{pri}(k) \Big] \Big\} \tag{31}$$

3.2.2. Action Space $\mathcal{A}$

Action determines the UAV's movement, including direction and speed:

$$\boldsymbol{a}(k) = \{a_l(k) = [v_l(k), \varphi_l(k)] : v_l(k) \epsilon [0, V_{MAx}], \varphi_l(k) = [0, 2pi), \forall l \in \mathcal{M}\ \}. \tag{32}$$

3.2.3. State Transition Probability $\mathcal{P}$

The state transition of $\boldsymbol{d}_{l,i}(k)$ is determined by $d_{i,l}(k) = \sqrt{\|\boldsymbol{q}_l(k) - \boldsymbol{w}_i(k)\|_2^2 + H^2}$ and the movement of the UAVs. The position relationship with other UAVs is determined by the movement of all UAVs. The distance between the UAVs and the origin is determined by the movement of the UAVs. Given the known positions of the UAVs, $C_l(k)$ and $\boldsymbol{D}_i(k)$ are determined by the offloading strategy GAS algorithm.

3.2.4. Reward $\mathcal{R}$

When the program is running, the environment provides feedback on the actions of the UAVs. This paper aims to minimize the maximum service time. In order to accomplish the overarching objective, it is imperative for each time slot to efficiently offload a substantial amount of data while minimizing energy consumption. Consequently, all of the UAVs

exhibit cooperative behavior, thereby enabling us to establish a unified form for the reward of each UAV as follows:

$$R(k) = \eta_1 \sum_{l=1}^{M} E_l^C(k) + \eta_2 \sum_{l=1}^{M} \sum_{i=0}^{N} D_{i,l}(k). \tag{33}$$

To satisfy Equations (5) and (6), we set a collision penalty and an out-of-bounds penalty as follows:

$$\Phi_l^c(k) = \begin{cases} \Gamma^c & \text{if (5) is not satisfied,} \\ 0 & \text{otherwise,} \end{cases} \tag{34}$$

$$\Phi_l^b(k) \begin{cases} \Gamma^b & \text{if (6) is not satisfied,} \\ 0 & \text{otherwise.} \end{cases} \tag{35}$$

So the final reward for the UAVs is as per the following:

$$\hat{R}_l(k) = R(k) + \Phi_l^c(k) + \Phi_l^b(k) \tag{36}$$

3.2.5. WMDDPG Algorithm

In order to find the optimal solution to the above problem, we utilized the Weighted-Strategy-Based Multi-Agent Deep Deterministic Policy Gradient (WMDDPG) algorithm. The algorithm is shown in Figure 4:

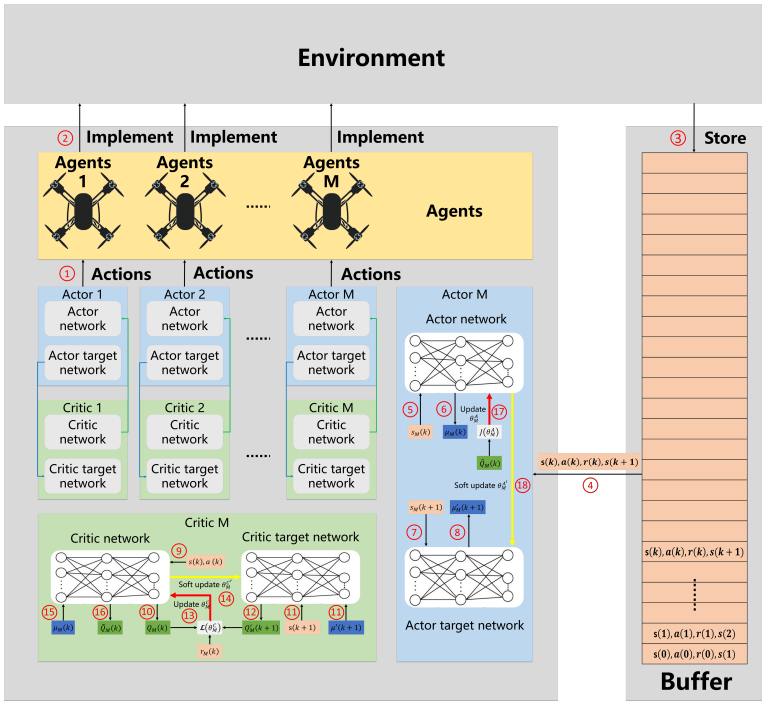

Figure 4. The WMDDPG algorithm.

First, the actor network and the critic network are initialized using parameters θ_M^C and θ_M^A. To prevent oscillation during training, target networks are introduced in both the actor and critic networks, with the parameters denoted as $\theta_M^{C'}$ and $\theta_M^{A'}$. To ensure network exploration, the Weight-Based Strategy algorithm or adding noise to the output of the actor network are employed to generate actions for each agent in the early stages of training (labeled as ① in Figure 4).

The Weight-Based Strategy (WS) algorithm: For each UAV, the weight of the different UE is a function of the distance from the UAV and whether there are data to be calculated. The UAV always moves toward the UE with the largest weight. Because the UAV can

serve multiple UE, the UE that are closer to each other can be regarded as the same UE. The pseudo-code for the WS algorithm is given in Table 2.

Table 2. The WS algorithm.

Algorithm WS algorithm.
1: Initialize epsilon = 1.0.
2: **if** random(0, 1) <= epsilon **then**
3: Update the distance between all UE and UAVs:$d_{i,l}(k)$.
4: Initialize the sorted list of distance between all UAVs and UE: distance_sort, and sort it in ascending order.
5: Initialize the list of selected UE in time slot k as the empty list: selected_UE= $[] * M$.
6: Initialize the list of each UAV for the time slot k to move towards the UE as the empty list: uav_move = $[] * M$.
7: **for** $l = 1, 2, \ldots, M$ **do**
8: **for** i in distance_sort$[l]$ **do**
9: **if** len(selected_UE$[l]$ ==0) and i not in selected_UE and the distance from the previously selected UE >= 80 **then**
10: add i to the list selected_UE$[l]$
11: **end if**
12: **end for**
13: **end for**
14: Initialize the sorted list of distances between the already selected UE and all UAVs: distance_selected_UE, and sort it in ascending order.
15: **for** $l = 1, 2, \ldots, M$ **do**
16: **for** i in distance_selected_UE$[l]$ **do**
17: **if** len(uav_move$[l]$ ==0) and i not in uav_move **then**
18: add i to the list uav_move$[l]$
19: **end if**
20: **end for**
21: **end for**
22:**end if**

The environment generates rewards and next states based on the actions deployed by each agent during the current time step, and it then stores them as $[s(k), a(k), r(k), s(k+1)]$ (labeled as ② and ③ in Figure 4). Once a sufficient number of different time steps are stored, the network training can commence.

The stored $[\mathrm{s}(k), a(k), r(k), s(k+1)]$ is then extracted into the network structure. Different actor networks take the respective agent's observation $s_l(k)$ as inputting and outputting the corresponding predicted actions $\mu_l(k)$ (labeled as ⑤ and ⑥ in Figure 4). Different actor target networks take the respective agent's observation $s_l(k+1)$ as inputting and outputting the corresponding predicted actions $\mu'_l(k+1)$ (labeled as ⑦ and ⑨ in Figure 4), which together form the predicted actions $\boldsymbol{\mu}'(k+1)$ for all agents at the next time step.

$$\boldsymbol{\mu}'(k+1) = \left\{ \left[\mu'_0(k+1), \mu'_1(k+1), \ldots, \mu'_M(k+1) \right] \right\}. \tag{37}$$

As shown in Figure 4, by inputting $\mathbf{s}(k)$ and $\mathbf{a}(k)$ into the critic network and replacing $a_l(k)$ in $\mathbf{a}(k)$ with $\mu_l(k)$, the corresponding score $\widetilde{Q}_l(k)$ under this condition is obtained. The parameters of the actor network are updated using policy gradients, as shown in step ⑰ in Figure 4.

$$\nabla_{\theta_M^C} J \approx \frac{1}{S} \sum_j \nabla_{\theta_M^C} \mu_l(s_l(k)) \nabla_{a_l(k)} \widetilde{Q}_l(\mathrm{s}(k), a_0(k), \ldots, a_l(k), \ldots, a_M(k)) \Big|_{a_l(k)=\mu_l(s_l(k))} \tag{38}$$

As shown in steps ⑨ and ⑩ in Figure 4, different critic target networks input $\mathbf{s}(k)$ and output scores $Q_l(k)$ for action $\mathbf{a}(k)$. As shown in steps ⑪ and ⑫ in Figure 4, different

critic networks input s$(k+1)$ and output scores $Q_l'(k+1)$ for action $\boldsymbol{\mu}'(k+1)$. In step ⑬ of Figure 4, the actual Q value is calculated using the Bellman equation.

$$\hat{Q}_l(k) = r_l(k) + \gamma Q_l'(k+1). \tag{39}$$

Based on the actual Q value and the estimated Q value, the loss function of the critic network can be calculated:

$$L\left(\theta_M^C\right) = \frac{1}{S} MSE\left(Q_l(k), \hat{Q}_l(k)\right). \tag{40}$$

Finally, in steps ⑭ and ⑱, the parameters of the target network are periodically updated according to a rule:

$$\theta_M^{A'} \leftarrow \tau_0 \theta_M^A + (1-\tau_0)\theta_M^{A'}, \tag{41}$$

$$\theta_M^{C'} \leftarrow \tau_0 \theta_M^C + (1-\tau_0)\theta_M^{C'}. \tag{42}$$

where $\boldsymbol{\tau}_0 \in (0,1)$ represents the soft update rate.

4. Simulated Result

In this section, we initially present the simulation configuration details and subsequently analyze the training performance of the proposed WMDDPG-GSA algorithm when using the Weight-Based Strategy (WS), Random Choice (RC) algorithm, and traditional MADDPG algorithm as evaluation benchmarks. During the testing phase, we validate the effectiveness of our algorithm by comparing it with the baseline algorithms in a simulated environment.

4.1. Simulation Settings

We consider a service area with a radius of 300 m, which encompasses 16 randomly distributed UE and 3 UAVs. The initial position of the UAV is fixed while maintaining a flight height of 50 m. Furthermore, the value for other environmental parameters are presented in Table 3.

Table 3. Simulation parameters.

Notation	Physical Meaning	Value
D_{max}	Coverage area of the UAVs	100 m
D_{min}	Minimum distance allowed between UAVs	15 m
Δ	Duration of time slots	1 s
V_{max}	Maximum flight speed	30 m/s
C_{max}	Number of UE that the DP-UAVs can compute in parallel	3
f^{uav}	Computing capacity of UAVs	3×10^9 cycle/s
f^{ue}	Computing capacity of the UE	1×10^7 cycle/s
B	Bandwidth	5×10^5 Hz

The actor and critic networks both feature a fully connected hidden layer comprising 64 neurons, with the Rectified Linear Unit (ReLU) serving as the activation function for this layer. The activation function of the output layer is the logical function (sigmoid). Further details regarding the parameter settings associated with neural network training can be found in Table 4.

Table 4. Training parameters.

Training Parameter	Value
Total number of training steps	1×10^5
Learning rate of critic network	5×10^{-5}
Learning rate of actor network	5×10^{-4}
Discount factor	0.9
Soft update rate for target networks	0.01

4.2. Learning Stage

The variations in the reward function during training for different UAVs, given the aforementioned parameters, are depicted in Figure 5. The cumulative reward shows an upward trend from the beginning of training and eventually stabilizes at approximately 25,000 after 40,000 steps. During the training process, a collision occurred between UAV2 and UAV3, thereby leading to a sudden decrease in the reward function. The model promptly adjusted its policy in subsequent training iterations, thus leading to a continuous rise in the reward function. To further verify the effectiveness of the WMDDPG-GSA algorithm, we propose three basic algorithms:

The Random Choice (RC) algorithm: The flight actions of the UAVs within the limited range are randomly selected.

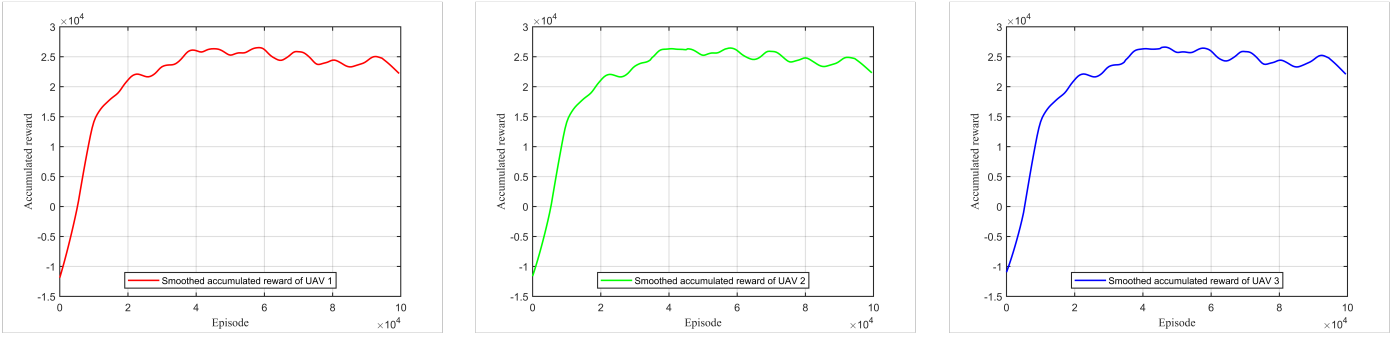

Figure 5. The cumulative reward of the UAVs.

The Weight-Based Strategy (WS) algorithm: The weights of the different UE associated with each UAV are determined based on their distance to the UAV and the availability of data for calculation; meanwhile, the UAV always moves toward the UE with the highest weight.

The MADDPG-GSA algorithm: The network parameters are the same as the WMDDPG-GSA algorithm, but the actions of each agent are generated by random selection in the early stage of training or by adding noise to the output of the actor network.

The reward function in the training process is then compared with three baseline algorithms, and the corresponding results are presented in Figure 6. Specifically, we conducted simulations using the WS algorithm and the RC algorithm on 100 randomly generated environments. The average objective function value obtained for the WS algorithm was found to be 21,340.74, while that for the RC algorithm was determined to be 20,355.01. The reward function of the WMDDPG-GSA algorithm initially increased and surpassed that of the WS and RC algorithms within the first 20,000 training steps. When comparing it to the MADDPG-GSA algorithm, we can observe that the MADDPG-GSA algorithm converges around 80,000 steps with significantly inferior convergence compared to the WMDDPG-GSA algorithm. This suggests that combining the WS algorithm with MADDPG can accelerate convergence speed and enhance convergence effectiveness.

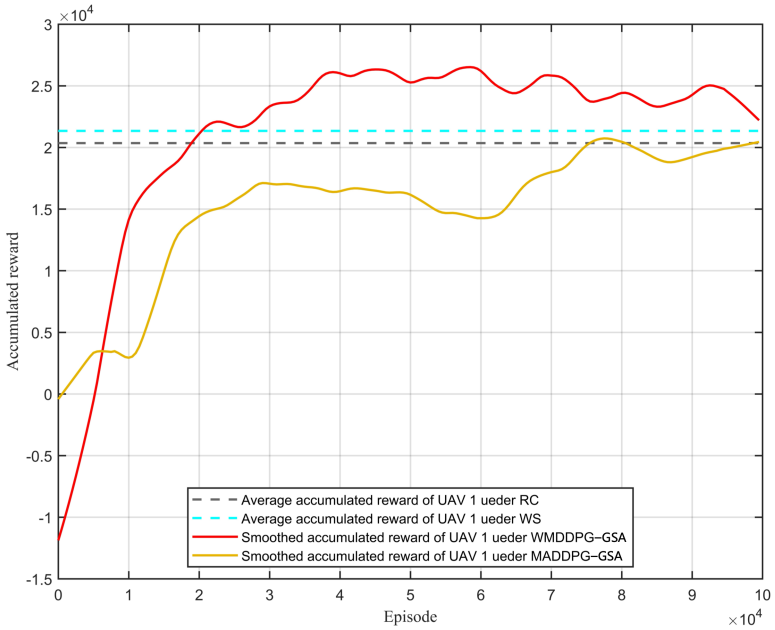

Figure 6. Comparison of the cumulative reward functions of different algorithms.

4.3. Testing Phase

After the agents complete the centralized training, the UAVs carry a network of pre-trained actors and service the UE based on their own private observations. To validate the effectiveness of the pre-trained actor network in practical applications, we conduct tests across multiple random environments, and plot the trajectory of the UAV in some environments in Figure 7. The initial position of the UAV is denoted by a cross, while its trajectory is represented by a line. The gray dots indicate the positions of the UE. In the trajectories of the UAVs, it is evident that they exhibit divergent flight paths while maintaining collision-free operations, thereby facilitating cooperative behavior. During the initial stage, UAVs tend to operate at higher speeds, thus enabling faster service delivery to the UE. Once one or more UE are connected, the UAVs transition into a hover or low-speed movement toward the location with the next highest priority.

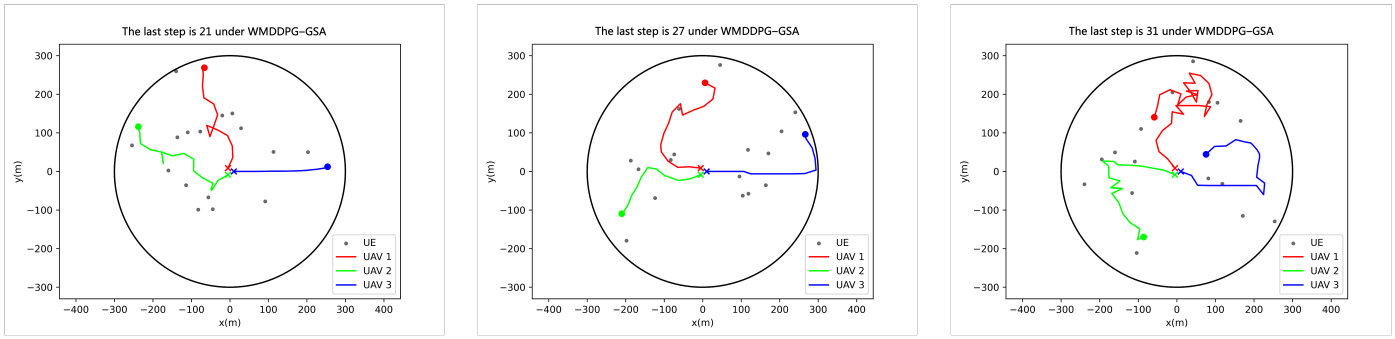

Figure 7. The trajectory of the UAVs in some environments.

The timeliness of the algorithm is primarily positively correlated with the aggregation degree of the UE in terms of efficiency. A higher aggregation degree results in improved timeliness, as it allows for a simultaneous service to a greater number of UE by the UAVs. In this scenario, under the condition of an uninterrupted power supply, all terminals perform data operations through local computing, thereby resulting in a total time of approximately 465 s. This paper employs the WMDDPG-GSA algorithm and achieves a task completion

time reduction of over 90% across various scenarios. In the test scenario, the best shortening effect reached 96.74%.

We conducted a test comparison of the WMDDPG-GSA algorithm with three benchmark algorithms. Some of the test results are shown in Figure 8, and it can be observed that the proposed algorithm outperforms the benchmark algorithms. Compared to the RC algorithm, the WMDDPG-GSA algorithm reduced the task completion time by 93.23%. Compared to the WS algorithm, the WMDDPG-GSA algorithm reduced the task completion time by 8.70%. Compared to the MADDPG-GSA algorithm, the WMDDPG-GSA algorithm reduced the task completion time by 22.22%. From the test results, it can also be inferred that the combination of the WS algorithm and the MADDPG-GSA algorithm has played a very instructive role in further reducing the task completion time of the WMDDPG-GSA algorithm proposed in this paper.

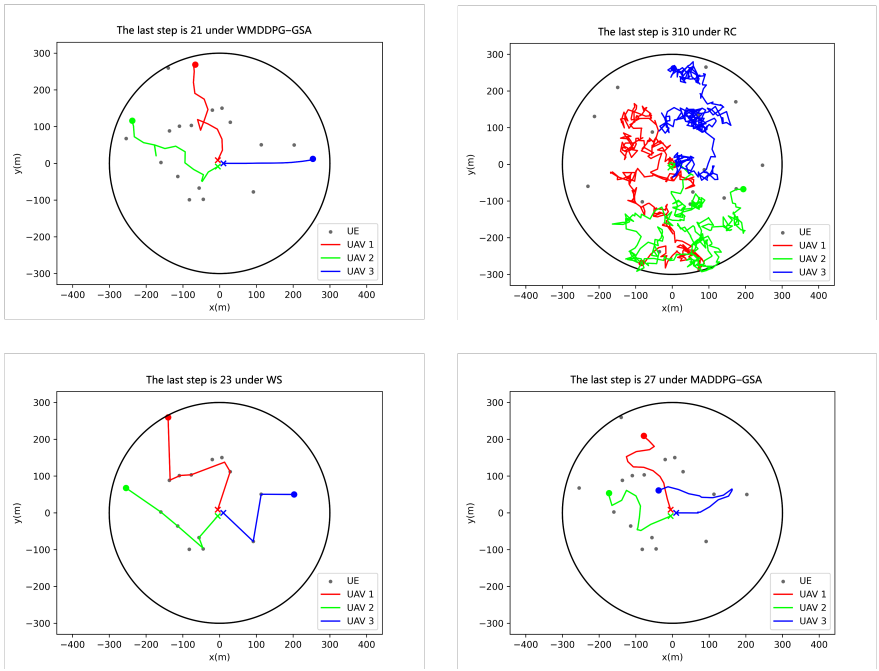

Figure 8. Test comparison of the different algorithms.

5. Conclusions

This study focuses on trajectory design and a task offloading strategy optimization in a UAV-assisted Mobile Edge Computing (MEC) system while considering energy constraints. To address this issue, we decomposed the problem into two subproblems, and we propose the Weighted-Strategy-Based Multi-Agent Deep Deterministic Policy Gradient (WMDDPG) algorithm for trajectory design and the Greedy-based Simulated Annealing (GSA) algorithm for task offloading strategy optimization. In order to account for the impact of the Doppler Frequency Shift (DFS) on UAV data transmission during flight, an estimation and compensation method was employed to mitigate its effects. The numerical results demonstrate that the proposed WMDDPG-GSA algorithm achieves a superior performance compared to benchmark algorithms such as the Weight-Based Strategy (WS), Random Selection (RC), and the conventional Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm.

Author Contributions: Methodology, T.M.; Software, T.M. and Y.Y.; Validation, T.M.; Writing-original draft, T.M.; Writing—review & editing, H.X. and T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Key Research and Development Program of Jiangsu Province (no. BE2020084-2) and the National Key Research and Development Program of China (no. 2020YFB1600104).

Data Availability Statement: No additional data are available.

Conflicts of Interest: No potential competing interests were reported by the authors.

References

1. Sun, L.; Wang, J.; Lin, B. Task Allocation Strategy for MEC-Enabled IIoTs via Bayesian Network Based Evolutionary Computation. *IEEE Trans. Ind. Inform.* **2021**, *17*, 3441–3449. [CrossRef]
2. Seng, S.; Yang, G.; Li, X.; Ji, H.; Luo, C. Energy-Efficient Communications in Unmanned Aerial Relaying Systems. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2780–2791. [CrossRef]
3. Zhou, Y.; Pan, C.; Yeoh, P.L.; Wang, K.; Elkashlan, M.; Vucetic, B.; Li, Y. Secure Communications for UAV-Enabled Mobile Edge Computing Systems. *IEEE Trans. Commun.* **2020**, *68*, 376–388. [CrossRef]
4. Luo, Z.; Qian, X.; Huang, C.; Ding, R.; Xin, N.; Su, D. Unmanned Collaborative Intelligent Edge Computing Task Scheduling. In Proceedings of the 2021 IEEE International Conference on Unmanned Systems (ICUS), Beijing, China, 15–17 October 2021; pp. 309–314. [CrossRef]
5. Zhan, C.; Zeng, Y. Completion Time Minimization for Multi-UAV-Enabled Data Collection. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 4859–4872. [CrossRef]
6. Belaoura, W.; Ghanem, K.; Shakir, M.Z.; Hasna, M.O. Performance and User Association Optimization for UAV Relay-Assisted mm-Wave Massive MIMO Systems. *IEEE Access* **2022**, *10*, 49611–49624. [CrossRef]
7. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Unmanned Aerial Vehicle With Underlaid Device-to-Device Communications: Performance and Tradeoffs. *IEEE Trans. Wirel. Commun.* **2016**, *15*, 3949–3963. [CrossRef]
8. Zhou, F.; Hu, R.Q.; Li, Z.; Wang, Y. Mobile Edge Computing in Unmanned Aerial Vehicle Networks. *IEEE Wirel. Commun.* **2020**, *27*, 140–146. [CrossRef]
9. Zhang, T.; Xu, Y.; Loo, J.; Yang, D.; Xiao, L. Joint Computation and Communication Design for UAV-Assisted Mobile Edge Computing in IoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 5505–5516. [CrossRef]
10. Gan, Y.; He, Y. Trajectory Optimization and Computing Offloading Strategy in UAV-Assisted MEC System. In Proceedings of the 2021 Computing, Communications and IoT Applications (ComComAp), Shenzhen, China, 26–28 November 2021; pp. 132–137. [CrossRef]
11. Argyriou, A. Passive Angle-Doppler Profile Estimation for Narrowband Digitally Modulated Wireless Signals. In Proceedings of the 2022 IEEE 12th Sensor Array and Multichannel Signal Processing Workshop (SAM), Trondheim, Norway, 20–23 June 2022; pp. 246–250. [CrossRef]
12. Liao, T.; Meng, Q.; Liu, C.; Zhang, M.; Li, B. Study on Doppler Frequency Offset Estimation Algorithm for High Speed Maglev. In Proceedings of the 2021 International Conference on Communications, Information System and Computer Engineering (CISCE), Beijing, China, 14–16 May 2021; pp. 755–758. [CrossRef]
13. Kang, E.S.; Hwang, H.; Han, D.S. A fine carrier recovery algorithm robustto doppler shift for OFDM systems. *IEEE Trans. Consum. Electron.* **2010**, *56*, 1218–1222. [CrossRef]
14. Zhang, Q.; Sun, H.; Feng, Z.; Gao, H.; Li, W. Data-Aided Doppler Frequency Shift Estimation and Compensation for UAVs. *IEEE Internet Things J.* **2020**, *7*, 400–415. [CrossRef]
15. Hu, X.; Wong, K.K.; Yang, K.; Zheng, Z. UAV-Assisted Relaying and Edge Computing: Scheduling and Trajectory Optimization. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 4738–4752. [CrossRef]
16. Wang, Y.; Gao, Z. Three-Dimensional Trajectory Design for Multi-User MISO UAV Communications: A Deep Reinforcement Learning Approach. In Proceedings of the 2021 IEEE/CIC International Conference on Communications in China (ICCC), Xiamen, China, 28–30 July 2021; pp. 706–711. [CrossRef]
17. Zhang, X.; Zhang, H.; Ji, H.; Li, X. Joint Optimization of UAV Trajectory and Relay Ratio in UAV-Aided Mobile Edge Computation Network. In Proceedings of the 2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications, London, UK, 31 August–3 September 2020; pp. 1–6. [CrossRef]
18. Qian, Y.; Wang, F.; Li, J.; Shi, L.; Cai, K.; Shu, F. User Association and Path Planning for UAV-Aided Mobile Edge Computing with Energy Restriction. *IEEE Wirel. Commun. Lett.* **2019**, *8*, 1312–1315. [CrossRef]
19. Ding, R.; Gao, F.; Shen, X.S. 3D UAV Trajectory Design and Frequency Band Allocation for Energy-Efficient and Fair Communication: A Deep Reinforcement Learning Approach. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 7796–7809. [CrossRef]
20. Liu, X.; Liu, Y.; Chen, Y. Reinforcement Learning in Multiple-UAV Networks: Deployment and Movement Design. *IEEE Trans. Veh. Technol.* **2019**, *68*, 8036–8049. [CrossRef]
21. Xiao, H.; Hu, Z.; Yang, K.; Du, Y.; Chen, D. An Energy-Aware Joint Routing and Task Allocation Algorithm in MEC Systems Assisted by Multiple UAVs. In Proceedings of the 2020 International Wireless Communications and Mobile Computing (IWCMC), Limassol, Cyprus, 15–19 June 2020; pp. 1654–1659. [CrossRef]
22. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *arXiv* **2020**, arXiv:1706.02275.

23. Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; Whiteson, S. Counterfactual Multi-Agent Policy Gradients. *arXiv* **2017**, arXiv:1705.08926.
24. Wang, L.; Wang, K.; Pan, C.; Xu, W.; Aslam, N.; Hanzo, L. Multi-Agent Deep Reinforcement Learning-Based Trajectory Planning for Multi-UAV Assisted Mobile Edge Computing. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *7*, 73–84. [CrossRef]
25. Wu, D.; Wan, K.; Tang, J.; Gao, X.; Zhai, Y.; Qi, Z. An Improved Method towards Multi-UAV Autonomous Navigation Using Deep Reinforcement Learning. In Proceedings of the 2022 7th International Conference on Control and Robotics Engineering (ICCRE), Beijing, China, 15–17 April 2022; pp. 96–101. [CrossRef]
26. Xu, W.; Zhang, T.; Yang, L. Joint Computing Offloading and Trajectory for Multi-UAV Enabled MEC Systems. In Proceedings of the 2022 IEEE 22nd International Conference on Communication Technology (ICCT), Nanjing, China, 11–14 November 2022; pp. 510–515. [CrossRef]
27. Zhang, Y.; Han, X.; Dong, Y.; Xie, J.; Xie, G.; Xu, X. A novel state transition simulated annealing algorithm for the multiple traveling salesmen problem. *J. Supercomput.* **2021**, *77*, 11827–11852. [CrossRef]
28. Choachaicharoenkul, S.; Coit, D.; Wattanapongsakorn, N. Multi-Objective Trip Planning With Solution Ranking Based on User Preference and Restaurant Selection. *IEEE Access* **2022**, *10*, 10688–10705. [CrossRef]
29. Zhu, B.; Bedeer, E.; Nguyen, H.H.; Barton, R.; Gao, Z. UAV Trajectory Planning for AoI-Minimal Data Collection in UAV-Aided IoT Networks by Transformer. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 1343–1358. [CrossRef]
30. Al-Hourani, A.; Kandeepan, S.; Lardner, S. Optimal LAP Altitude for Maximum Coverage. *IEEE Wirel. Commun. Lett.* **2014**, *3*, 569–572. [CrossRef]