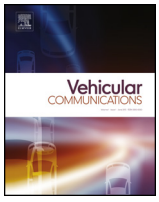
Vehicular Communications 43 (2023) 100632

Contents lists available at ScienceDirect

Vehicular Communications

journal homepage: www.elsevier.com/locate/vehcom



Joint optimization of trajectory, beamforming, and power allocation in UAV-enabled WPT networks using DRL combined with water-filling algorithm ☆

Sengly Muy, Jung-Ryun Lee *

School of Electronics and Electrical Engineering (EEE) and Department of Intelligent Energy and Industry at Chung-Ang University, Seoul, 156-756, Republic of Korea

ARTICLE INFO

Article history:
Received 25 July 2022
Received in revised form 15 March 2023
Accepted 12 June 2023
Available online 29 June 2023

Keywords:
UAV's trajectory
WPT
Deep Q-learning

ABSTRACT

In this work, we study a wireless power transfer (WPT) network supported by an unmanned aerial vehicle (UAV), in which a UAV equipped with an array of antennas supplies wireless energy to charge ground user (GU) devices. With this configuration, we aim to maximize the lowest GU energy by jointly optimizing the UAV's trajectory, beamforming pattern, and transmitting power. Because the proposed maximization of the lowest GU energy is a non-convex optimization problem that is difficult to be solved, we transform the problem into a grid world problem in discrete-time space and propose a design of a deep reinforcement learning (DRL) approach that optimizes the loss function by determining the UAV's moving direction, beamforming angle, and transmit power level. Furthermore, we combine the water-filling algorithm with DRL, which can assist DRL in determining the hovering duration. Simulation results verify that our proposed design increases the minimum energy of GUs received from the UAV compared to the successive hover-and-fly algorithm while exploring the most appropriate path with low computational complexity.

1. Introduction

Researchers are intrigued by the potential of radio-frequency energy harvesting via wireless power transfer (WPT) as a means of supplying low-power Internet-of-things (IoT) devices with suitable and reliable energy sources. Moreover, WPT also has the ability to charge multiple wireless devices simultaneously, regardless of their movement or complex deployment, via multiple beamforming of array antennas [21]. Multiple-beamforming array antennas have proven helpful in a range of applications, including geostationary and aerospace telecommunications, due to their performance and flexibility [2]. However, the available infrastructure may not be able to enable WPT services for wireless IoT devices that are located in disaster areas or other situations where infrastructure is malfunctioning.

Recently, unmanned aerial vehicles (UAVs) have brought many benefits from their applications, such as remote sensing, search and rescue operations, goods delivery, security/surveillance, agriculture monitoring, and civil infrastructure inspection [11]. Additionally, UAVs are simple to deploy, low-cost, require less maintenance, and can be adapted to various environmental conditions. For these reasons, deploying UAVs to enable WPT services for wireless IoT devices is one of the best solutions for IoT devices being situated in disaster areas or failing infrastructure. That said, the flight duration of a UAV is limited by the onboard battery capacity, so optimizing the efficient power consumption of UAVs is an additional problem that needs to be solved [34,1,30]. However, it is difficult to optimize UAV trajectory planning with system performance using conventional optimization techniques because the problem must satisfy terrain and threat avoidance along with performance constraints.

Reinforcement learning (RL) algorithms have recently become popular in wireless networks for optimizing various problems and improving system performance while increasing the intelligence of edge devices. RL is designed to maximize a reward function by investigating the action domain through trial-and-error interactions

☆ This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC support program (IITP-2023-2018-0-01799) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation), by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (No. 20214000000280), and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. RS-2023-00251105).

* Corresponding author at: Chung-Ang University, 84 Heuksuk-ro, Dongjak-gu, Seoul, Republic of Korea.

E-mail addresses: muysengly@cau.ca.kr (S. Muy), jrlee@cau.ac.kr (J.-R. Lee).

https://doi.org/10.1016/j.vehcom.2023.100632
2214-2096/

in order to explore the best decision [26]. Q-learning (QL) is a powerful RL algorithm that can learn the value of an action in a specific state and aims to achieve a solution that is close to the global-optimal [6]. However, as the data size in the Q-table (state, action, and Q-value) in QL increases, excessive memory is required during the training process, which is known as the curse of dimensionality. Meanwhile, deep Q-learning (DQL), a form of deep reinforcement learning (DRL), uses a deep neural network (DNN) as a function approximator to handle raw input data with high dimension [20]. Because DQL shows its effectiveness in dealing with various problems with high complexity [13,14,10,9], it has been used for solving trajectory planning and resource allocation issues in UAV-assisted wireless networks [3].

2. Related work

Over the last decade, many works have been conducted to evaluate the effectiveness of deploying UAV-enabled WPT by effectively managing and supervising the UAV's trajectory while ensuring optimal resource allocation. A UAV-enabled mobile edge computing (MEC) system was studied by the authors of [7], in which a UAV first charges the IoT devices via WPT, and then each IoT device transmits the acquired data to the UAV by using the energy harvested from the UAV. In [27], the authors investigated the deployment of a UAV-enabled WPT network to charge the GUs by downlink. The GUs can use the harvested energy to send independent data to the UAV via uplink, while the target of the work is to ensure maximizing the minimum uplink throughput with all GUs over a limited UAV flight period. The authors of [32] presented a new successive hover-and-fly algorithm to optimize the UAV's suitable trajectory, in which a specific set of locations for efficient energy transmission is determined, and the UAV flies between those locations and hovers only at those locations for efficient energy transmission. The authors of [28] studied the maximization of the amount of energy harvesting of all GUs by optimizing the trajectory plan of a UAV. Assuming an ideal case, the authors ignored the UAV's maximum velocity constraint and used the Lagrange dual function to get the best possible trajectory. As a next step, an alternative successive hover-and-fly algorithm using the successive convex programming optimization method was proposed to obtain the practical trajectory design.

Recently, machine learning (ML) techniques have been applied to WPT systems to optimize their efficiency and performance. The author of [24] proposed an integrated block-chain and multi-agent DRL for solving the optimization problem of computation offloading and energy harvesting (EH) while considering the optimal resource price. The author of [17] proposed a design deep Q-network (DQN) in order to address the problem of minimizing the average age of information (AoI) of the ground nodes (GNs) by jointly optimizing the trajectory of the UAV and the scheduling of information transmission and energy harvesting of GNs. The author of [36] proposed a deep deterministic policy gradient (DDPG) with the goal of maximizing the amount of data offloaded to the UAV. In this system, the UAV harvests the energy from the base station and assists the edge server with the vehicle's computing tasks.

It is noted that previous studies focused on deploying the UAV to maximize energy efficiency, energy transfer, communication coverage, or sum rate of GUs. In our work, we focus on the max-min problem for the residual GU's energy, which means maximizing the minimum residual battery capacity of a GU in a network. To be more specific, a UAV with multiple-beamforming array antennas is deployed in a network to transfer wireless energy and charge the batteries of ground-based IoT devices. With this network scenario, we construct the optimization problem of maximizing the lowest GU energy by considering the UAV's control parameters, including the trajectory, hovering duration, beamforming pattern, and transmitting power. The following outlines summarize the main contributions of our paper. 1) To simplify the problem, we convert the UAV's flying time from continuous to discrete format, and we also quantize the UAV's position in the grid world problem, which is the first problem that is faced by DRL. 2) We propose a DQL design based on the water-filling algorithm to optimize the grid world problem, in which UAV placement, hovering duration, beamforming pattern, and power allocation are jointly optimized. In this case, the water-filling algorithm aids in the reduction of DQL training, which determines the UAV's hovering duration. Moreover, a multivariate normal distribution's probability density function is used to design the reward value at each grid. 3) Simulation results demonstrate that our new approach outperforms the previous successive hover-and-fly algorithm with low computational time.

The remaining of this paper is arranged as follows. The system model and optimization of a UAV-enabled WPT network with multiple-beamforming array antennas to dispatch energy to IoT GUs are described in Section 2. In Section 3, we present the proposed design of the DQL with the water-filling algorithm to solve the grid world problem for a UAV. Sections 4 and 5 demonstrate the simulation results and the conclusion of this paper, respectively.

3. System model and problem formulation

3.1. System model

We consider a UAV-supported WPT network, where K GUs are deployed in a cell with a radius of R and a charging period of T. The set of GUs is denoted as $\mathcal{K} = \{1, 2, ..., K\}$ and the available UAV flying time is expressed as $\mathcal{T} = (0, T]$. Using the Cartesian coordinate system, we assume that all GUs are positioned on the ground at point $(x_k, y_k, 0), \forall k \in \mathcal{K}$. On the other hand, the UAV's location at time t is denoted as $(x(t), y(t), H), \forall t \in \mathcal{T}$ with a fixed altitude H.

Here, we assume that the altitude of the UAV is fixed due to simplicity, mission requirement, system constraints, and cost reduction issues [23,22,25]. Let V denoted as the maximum speed of the UAV, and thus the flight constraint for the UAV is given by

$$x^2(t) + y^2(t) \leq R^2, \forall t \in \mathcal{T}, \tag{1}$$

$$\dot{x}^2(t) + \dot{y}^2(t) \leq V^2, \forall t \in \mathcal{T}, \tag{2}$$

where $\dot{x}(t)$ and $\dot{y}(t)$ are denoted as the time-derivative of $x(t)$ and $y(t)$, respectively.

In this network scenario, we apply the line-of-sight (LOS) environment so as to simplify the channel model and allow for easier analysis of the system's performance assuming a flat surface and omitted obstacles (such as trees, buildings, etc.), [29]. Therefore, the LOS path predominates over the non-line-of-sight (NLOS) paths, [5]. Moreover, we consider the power attenuation model as the path loss [35], which means the reduction in power density (attenuation) of an electromagnetic wave as it propagates through space or through a medium such as air, water, or a solid material. In this work, we consider a channel model following the propagation loss (path loss) with the steering beam [8,33,37].

Therefore, the channel model from the UAV to the GU k can be expressed as, [8,33,37]:

$$\mathbf{h}_k = \sqrt{\beta_0 d_k^{-\alpha}}\,\mathbf{a}(\theta, \phi), \tag{3}$$

where α is the path-loss exponent, and β_0 is the channel power at the reference distance $d_0 = 1$ m. d_k is the distance from the UAV to GU k, which can be calculated as

$$d_k = \sqrt{(x - x_k)^2 + (y - y_k)^2 + H^2}. \tag{4}$$

Here, we assume that all GUs are capable of absorbing very low wireless power to charge their batteries based on the concept of long-range WPT, which is the invention of high power microwave amplifiers [18]. As a result, the maximum distance between the drone and the GUs can be calculated as

$$d_k \leq H \cos \Theta_{\max}, \forall k \in \mathcal{K}, \tag{5}$$

where $\Theta_{\max}$ is the maximum elevation angle. And, $\mathbf{a}(\theta, \phi)$ is the steering vector of elevation angle θ and azimuth angle ϕ for the LOS path, which is defined as

$$\mathbf{a}(\theta, \phi) = \Big[1, ..., e^{j\frac{2\pi}{\lambda} d_s \sin(\theta)[(a-1)\cos(\phi)+(b-1)\sin(\phi)]}, ..., e^{j\frac{2\pi}{\lambda} d_s \sin(\theta)[(A-1)\cos(\phi)+(B-1)\sin(\phi)]}\Big]^T \tag{6}$$

where λ is the wavelength, and d_s is the spacing between antenna elements. Furthermore, the effective channel gain from UAV to GU k is given by

$$\left|\mathbf{h}_k^H \boldsymbol{\upsilon}\right|^2 = \frac{\beta_0}{d_k^\alpha} \left|\mathbf{a}(\theta, \phi)\, \boldsymbol{\upsilon}\right|^2 \tag{6}$$

where $\mathbf{E}(\theta, \phi) = \mathbf{a}(\theta, \phi)\, \boldsymbol{\upsilon}$ is the beamforming pattern of the antenna array with the size of $A \times B$ antennas. It is noted that the direction of the main lobe can be controlled by linearly adjusting the beamforming vector $\boldsymbol{\upsilon}$.

In this study, we consider a uniform planar array antenna installed at the bottom of the UAV, which generates multiple independent steered beams. A uniform planar array antenna is divided into many sub-array antennas, and each sub-array antenna creates an independently directed beamforming by altering its phase shifters. We assume that the UAV is also installed with the global positioning system (GPS) sensor that is presumably used to identify the direction of the various guided beams based on the location of the GUs. In this setup, the array factor (AF) and beamforming pattern for an $A \times B$ array antennas are expressed as

$$AF = \sum_{a=1}^{A} \sum_{b=1}^{B} I_{a,b} \times e^{j\psi(\theta,\phi)}, \tag{7}$$

$$\mathbf{E}_{\theta,\phi} = \sum_{a=1}^{A} \sum_{b=1}^{B} p_{a,b}(\theta, \phi)\, I_{a,b} \times e^{j\psi(\theta,\phi)}, \tag{8}$$

$$\text{for } \psi(\theta, \phi) = \frac{2\pi}{\lambda} d_s \sin(\theta) \times \left[(a-1)\cos(\phi) + (b-1)\sin(\phi) + \beta_{a,b}\right]. \tag{9}$$

Here, $p_{a,b}(\theta, \phi)$ is the active element pattern, $I_{a,b}$ is amplitude excitation, and $\beta_{a,b}$ is the progressive phase shift of the (a, b)-th array element, where $a \in \{1, 2, \ldots, A\}$ and $b \in \{1, 2, \ldots, B\}$. The main beamforming of the antenna array toward a specific direction is accomplished through the gradual phase shift using the phase shifters. $\boldsymbol{\upsilon} = \left[\upsilon_{1,1}, ..., \upsilon_{1,b}, ..., \upsilon_{a,b}, ..., \upsilon_{A,B}\right]^T$ is the beamforming vector which represents the controlling parameter of the amplitude and phase of each antenna element, $\upsilon_{a,b} = p_{a,b}(\theta, \phi)\, I_{a,b} \times e^{j\beta_{a,b}}$. Assuming that the UAV's transmit power control is p, the received radio frequency power by GU k is given by

$$Q_k = p\left|\mathbf{h}_k^H \boldsymbol{\upsilon}\right|^2 = \frac{p\beta_0 \left|\mathbf{E}_{\theta,\phi}\right|^2}{d_k^{\alpha/2}}. \tag{10}$$

With the received radio frequency power in (10), we can formulate the total energy harvesting received by each GU k over the whole charging time t as

$$EH_k(t) = \int_0^t Q_k(\tau)\, d\tau, \ \forall t \in \mathcal{T}. \tag{11}$$

3.2. Problem formulation

Our study aims to maximize the minimum received energy of all GUs by jointly optimizing the trajectory $(x(t), y(t))$, transmit power $p(t)$, and beamforming pattern $\mathbf{E}_{\theta,\phi}(t)$ of the UAV. Therefore, we can formulate the optimization problem as

$$(\text{P1}): \max_{(x(t),y(t)),p(t),\mathbf{E}_{\theta,\phi}(t)} \min_{k\in\mathcal{K}} E_k(t) \tag{12a}$$

$$\text{s.t. } 0 \leq \phi \leq 2\pi, \tag{12b}$$

$$0 \leq \theta \leq \Theta_{\max}, \tag{12c}$$

$$0 \leq p(t) \leq P_{\max}, \forall t \in \mathcal{T}, \tag{12d}$$

$$E_{\text{UAV}}(t) \leq E_{\max}, \forall t \in \mathcal{T}, \tag{12e}$$

$$x^2(t) + y^2(t) \leq R^2, \forall t \in \mathcal{T}, \tag{12f}$$

$$\dot{x}^2(t) + \dot{y}^2(t) \leq V^2, \forall t \in \mathcal{T}, \tag{12g}$$

where $P_{\max}$, and $E_{\max}$ are the maximum transmit power of the UAV, and the maximum UAV battery level, respectively. The above optimization problem is non-convex because of the instantaneous locations of both the UAV and the GUs. Moreover, in the scenario considered, both the large- and small-scale fading are complicated to resolve directly using traditional optimization methods.

4. Proposed DRL with the water-filling algorithm

In this section, we present the deep Q-Learning (DQL) with the water-filling algorithm for UAV trajectory design and resource allocation.

4.1. Preliminaries

4.1.1. Reinforcement learning (RL)

Mathematically, RL problems are defined as four-tuple Markov decision processes (MDPs) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$. In particular, $\mathcal{S}$ represents the state space, $\mathcal{A}$ represents the action space, $\mathcal{P}$ represents the transition probability, and $\mathcal{R}$ represents the reward function. The transition probability $\mathcal{P}$ is the probability of transition from current state $s_t \in \mathcal{S}$ to the next state $s_{t+1} \in \mathcal{S}$ by an action $a_t \in \mathcal{A}$. The reward $\mathcal{R}$ is the return value of executing an action $a_t \in \mathcal{A}$ at state $s_t \in \mathcal{S}$, denoted as $r(s_t, a_t)$. To be more precise, an RL agent examines the environment and then determines its current state $s_t \in \mathcal{S}$. After selecting and carrying out an action $a_t \in \mathcal{A}$, the agent's next state $s_{t+1} \in \mathcal{S}$ can be acquired. At the end of each cycle, the agent is given a reward $r_t \in \mathcal{R}$ that is based on the environment. RL is intended to discover the policy $\pi : \mathcal{A} \times \mathcal{S} \to [0, 1]$, $\pi(a, s) = \Pr(a_t = a | s_t = s)$ for maximizing the accumulated expected reward.

4.1.2. Q-learning

Q-learning (QL) is a common and straightforward RL technique that focuses on determining the optimal action for a given current state using the Q-value. The Q-value produced by the state-action pair in QL during the trial-and-error test is recorded in the memory storage called Q-table. And the Q-table is updated by the

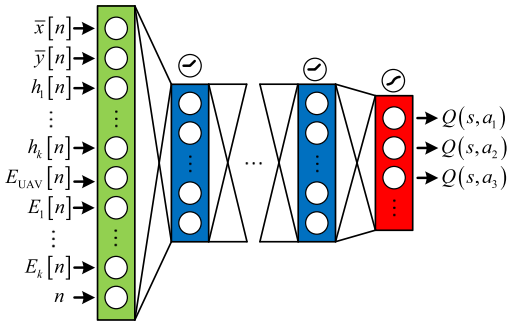


Fig. 1. The DQN design.

Q-function that is the core of the Bellman equation, which is expressed as

$$Q\left(s_{t+1}, a_{t+1}\right) = Q\left(s_t, a_t\right) + \alpha\left[r_t + \gamma \max_{a' \in \mathcal{A}} Q_t\left(s_{t+1}, a'\right) - Q\left(s_t, a_t\right)\right], \tag{13}$$

where $\alpha \in (0, 1]$ and $\gamma \in [0, 1]$ are the learning rate and the discount factor, respectively. It has been proved that QL can converge to Q^* when the discrete state and action spaces are finite [19].

4.1.3. Deep Q-learning

In our model, UAV flies flexibly in a 2D region and controls the power allocation and beamforming pattern; consequently, the Q-table requires a massive amount of memory, which causes a long computation time to discover the Q-value. By using DNN, Q-function is approximated as $Q(s, a) \approx Q(s, a, \boldsymbol{w})$, where $\boldsymbol{w}$ is the weight of DNN. DQL is trained in a way to maximize the loss function, which is given by

$$L(\boldsymbol{w}) = E\left[\left(y_t - Q(s, a, \boldsymbol{w})\right)^2\right], \tag{14}$$

where y_t is the target Q-value denoted as

$$y_t = r\left(s_t, a_t\right) + \lambda \max_{a'} Q_t\left(s', a', \boldsymbol{w}\right). \tag{15}$$

4.2. Proposed DQL with the water-filling algorithm

4.2.1. DQL design

Because the DQL algorithm is more effective for low-dimensional discrete action spaces, we convert the problem (12a) into discrete time-space by discretizing the whole charging duration into a finite number N of each time slot with duration $\delta = \frac{T}{N}$. It should be noticed that the duration δ is selected to be sufficiently small so that we may assume the UAV location remains roughly constant over each time slot n. We denote $x[n], y[n]$ as the position of UAV at time slot n, $p[n]$ as the UAV's transmit power at time slot n, and $\mathbf{E}_{\theta,\phi}[n]$ as the beamforming pattern at time slot n, where $n \in \mathcal{N} = \{1, 2, ..., N\}$. Therefore, we can formulate the optimization problem as

$$(\text{P2}): \max_{(x[n], y[n]), p[n], \mathbf{E}_{\theta,\phi}[n] k \in \mathcal{K}} \min E_k[n] \tag{16a}$$

$$\text{s.t. } 0 \le \phi \le 2\pi, \tag{16b}$$

$$0 \le \theta \le \Theta_{\max}, \tag{16c}$$

$$0 \le p[n] \le P_{\max}, \forall n \in \mathcal{N}, \tag{16d}$$

$$E_{\text{UAV}}[n] \le E_{\max}, \forall n \in \mathcal{N}, \tag{16e}$$

$$x^2[n] + y^2[n] \le R^2, \forall n \in \mathcal{N}, \tag{16f}$$

$$\Delta x^2[n] + \Delta y^2[n] \le V^2, \forall n \in \mathcal{N}, \tag{16g}$$

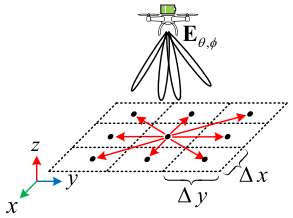


Fig. 2. The actions of UAV.

where $\Delta x^2[n] = (x[n] - x[n-1])^2$ and $\Delta y^2[n] = (y[n] - y[n-1])^2$.

To solve the optimization problem (P2) using DRL, we simplify (P2) via the grid world problem, which is well-known as the most basic and classic problem in reinforcement learning. To do so, we need to slice the location (x and y-axis) of the UAV into M^2 squares. We denote the location of the UAV in the grid world as $\bar{x}[n]$ and $\bar{y}[n]$, where $\bar{x}[n], \bar{y}[n] \in \left\{0, \frac{1}{M-1}2R, \frac{2}{M-1}2R, \ldots, 2R\right\}$. To find the UAV's trajectory in the grid world environment, we propose a design of a DQL model (state, action, and reward) as follows.

State: In our DRL design, the state is designed as

$$s[n] \in \left\{\left(\bar{x}[n], \bar{y}[n]\right), h_k[n], E_{\text{UAV}}[n], E_k[n], n\right\}_{k \in \mathcal{K}}, \tag{17}$$

where $h_k[n]$ is the channel gain from UAV to GU k at time slot n. $E_{\text{UAV}}[n]$ and $E_k[n]$ are the remaining battery of the UAV and the battery of ground devices k at time slot n, respectively.

Action: The direction of motion, transmit power, and beamforming pattern of the UAV are controlled by a user, and the action of DRL is defined as

$$a[n] \in \left\{\left(\Delta x[n], \Delta y[n]\right), p[n], \boldsymbol{E}_{\theta,\phi}[n]\right\} \tag{18}$$

where $(\Delta x[n], \Delta y[n])$ is the direction of motion of the UAV in the grid world following x- and y-axis at time slot n, which is given by

$$\Delta x[n] \in \left\{\bar{x}[n] - \frac{1}{M-1}2R, \bar{x}[n], \bar{x}[n] + \frac{1}{M-1}2R\right\}, \tag{19}$$

and

$$\Delta y[n] \in \left\{\bar{y}[n] - \frac{1}{M-1}2R, \bar{y}[n], \bar{y}[n] + \frac{1}{M-1}2R\right\}. \tag{20}$$

Therefore, in the grid world, the UAV has $(A_q = 9)$ different ways to move, such as: up, down, left, right, up-left, up-right, down-left, down-right, and not move. The moving direction of the UAV can be illustrated in Fig. 2. $0 \le p[n] \le P_{\max}$ is the UAV's transmit power with P_q quantized level at time slot n. $\boldsymbol{E}_{\theta,\phi}[n]$ is the beamforming pattern that is generated by E_q blocks of array antennas with elevation angle θ and azimuth angle ϕ at time slot n.

Reward: In DRL, the reward signal is used to evaluate how good an action is under a state. We notice that the lowest GU battery is the highest priority for the UAV to fly over or hover; therefore, we proposed a design of the reward function as

$$r[n] = \begin{cases} \sum_{k \in K} \bar{f}_{(x_k, y_k)}(x, y)\left(1 - \bar{E}_k[n]\right) & \text{satisfy (16e), (16f), and (16g),} \\ 0 & \text{otherwise,} \end{cases} \tag{21}$$

where $\bar{E}_k[n] = \frac{E_k[n] - \min_k E_k[n]}{\max_k E_k[n] - \min_k E_k[n]}$ is the normalized energy of the GU k. $\bar{f}_{(x_k, y_k)}(x, y) = \frac{f_{(x_k, y_k)}(x, y)}{f_{(x_k, y_k)}(x_k, y_k)}$ is the normalized probability density function (PDF) of multivariate normal distribution, given in the general case by

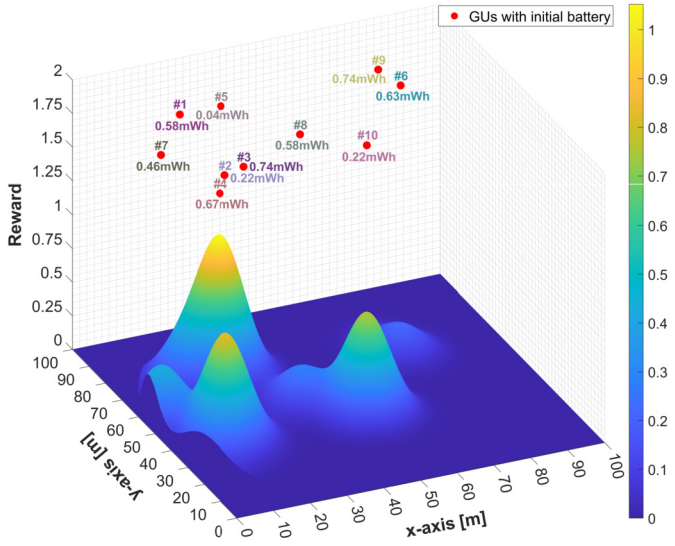


Fig. 3. The example of the reward function.

$$f_{\vec{\mu}}\left(\vec{x}\right) = \frac{\exp\left(-\frac{1}{2}\left(\vec{x}-\vec{\mu}\right)^T \mathbf{\Sigma}^{-1}\left(\vec{x}-\vec{\mu}\right)\right)}{\sqrt{(2\pi)\,|\mathbf{\Sigma}|}}, \tag{22}$$

where $\vec{\mu}$ is the mean vector, and $\mathbf{\Sigma}$ are co-variance matrix. $|\mathbf{\Sigma}| \equiv \det\Sigma$ is the generalized variance that is the determinant of Σ. Here, the group having GUs with low battery levels generates the highest reward area. Fig. 3 shows an example of the reward function with ten random GUs' locations and battery levels. Notice that the reward function provides the largest reward for the GU with the lowest battery capacity, and the reward value is changed based on the charging time.

DQN design: In our work, we design the fully connected DQN with zero bias, where the DQN consists of three layers, including an input layer, an output layer, and L hidden layers. The input layer is designed as the state, namely important information such as UAV location $(\bar{x}[n], \bar{y}[n])$ in the grid world environment, channel gain from UAV to GUs $(h_k[n])$, UAV's battery $(E_{\text{UAV}}[n])$, GUs' battery $(E_k[n])$, and the time slot n. In DQN, there are L hidden layers. Each layer uses a ReLu function, given as $f(x) = \max\{0, x\}$, as the activation function. Usually, DQN is used to approximate the Q-value; therefore, the output layer of DQN has to be designed as the Q-value-actions, and the Softmax function is used to activate this layer. The Softmax function is given as $f(x_i) = \frac{\exp(x_i)}{\sum_i \exp(x_i)}$.

Policy: An ϵ-greedy policy in DRL is a simple method to balance exploration and exploitation by choosing between exploration and exploitation randomly, which is given by

$$a[n+1] = \begin{cases} \underset{a'\in\mathcal{A}}{\text{argmax}}\, Q\left(s, a', \boldsymbol{w}\right), & \text{with prob. of } 1-\epsilon \\ \text{random action}, & \text{with prob. of } \epsilon. \end{cases} \tag{23}$$

4.2.2. The water-filling algorithm

The water-filling algorithm is a process of determining equalization strategies for different systems' performances. As the algorithm's name implies, water finds its level even when filled in one portion of a vessel with several openings due to Pascal's law. Thus, the water-filling algorithm is a practical algorithm to solve the max-min problems. To apply the water-filling algorithm to the proposed DRL, we first assume that there are P hovering locations, and denoting $p \in \mathcal{P} \triangleq \{1, 2, ..., P\}$ as the set of hovering points. Then the optimal hovering duration $\{T_1, T_2, ..., T_P\}$ can be calculated by the water-filling algorithm, where $h_{p,k}[n]$ is the channel from the p-th hovering point to the k-th GU at time slot n, and $(x, y, H)_p$ is the hovering point p. The pseudo-code for the application of the water-filling algorithm to the proposed DRL algorithm is given in Algorithm 1.

Algorithm 1 Hovering duration based on the water-filling algorithm.

output T_p
initialize:
  Randomly initialize the GUs' battery E_k
  Set $(x, y, H)_p = (x, y, H)_k$, $N = \frac{T}{\Delta}$, $T_p = 0$, E_{UAV}, $n = 1$
while $(n \le N$ and $E_{\text{UAV}} \ge 0)$
  $k^* = \underset{k\in\mathcal{K}}{\text{argmin}}\ E_k[n]$
  $p^* = \underset{p\in\mathcal{P}}{\text{argmin}}\ h_{p,k^*}[n]$
  for k in LoS of $(x, y, H)_{p^*}$
    $E_k[n] \leftarrow E_k[n] + \eta P_{\max}\delta h_{p^*,k}[n]$
  end for
  $E_{\text{UAV}}[n] \leftarrow E_{\text{UAV}}[n] - P_{\max}\delta$
  $T_{p^*} \leftarrow T_{p^*} + \delta$
  $n \leftarrow n + 1$
end while

The complete algorithm to maximize the lowest GU energy problem by jointly optimizing the UAV's trajectory, beamforming pattern, and transmit power for UAV enabled WPT network with the DQL technique is summarized in Algorithm 2.

Algorithm 2 DQL with the water-filling algorithm.

initialize
  Randomly initialize the location of GUs $(x, y)_k$
  Randomly initialize the Q-network $Q(s, a; \boldsymbol{w})$
  Randomly initialize the GUs' battery E_k
  Initialize the replay memory D with capacity C
  Initial the location of UAV to (R, R, H)
for each epoch **do**
  for each time slot n **do**
    the UAV gets the GUs' locations and formulated the state $s[n]$
    the agent takes action $a'[n]$ using ϵ-greedy policy via $Q(s, a, \boldsymbol{w})$
    if $v[n] = 0$ (The UAV takes an action that it is not move)
      set $(x[n], y[n], H)$ as a hovering location $(x, y, H)_p$
    end if
    Update to the state $s[n+1]$ and observe the reward $r[n]$
    Store $(s[n], a[n], r[n], s[n+1])$ in replay memory D
  end for
  Find the optimal hovering duration based on **Algorithm 1**
  if the replay memory D is full **then**
    Train the DQN
  end if
end for

4.3. Time complexity analysis

4.3.1. The successive hover-and-fly

The time complexity of the successive hover-and-fly algorithm is determined by the number of variables and the associated calculation of the objective functions [4]. According to the complexity analysis in [4], the ellipsoid method to solve a convex subproblem with $\Upsilon = A_q + P_q + E_q$ variables can be applied by using optimization-based iteration techniques (i.e. gradient descent with barrier function algorithm) that correspond to $\mathcal{O}\left(\Upsilon^2 \times \log\left(\frac{1}{\varepsilon}\right)\right)$ computation complexity, where the error threshold is $\varepsilon > 0$. Moreover, there are a total of $\mathcal{O}\left(R_q^2\right)$ possible testing candidates in the exhaustive 2D search of the successive hover-and-fly algorithm, where R_q is the quantized level along the x- and y-axes. Due to the ellipsoid updates [4] and solving (P1) based on the analysis in [15], the complexities of $\mathcal{O}\left(\Upsilon^2\right)$ and $\mathcal{O}\left(\Upsilon \times R_q\right)$ are introduced in each iteration. Therefore, the total time complexity of the successive hover-and-fly algorithm is given by

Table 1
Time complexity comparison.

Algorithm	Computation complexity	Operations
Successive hover-and-fly	$\mathcal{O}\left(\Upsilon^3 \times R_q^2 \times \log\left(\frac{1}{\varepsilon}\right) \times \max\left\{\Upsilon, R_q\right\}\right)$	1.21×10^{17}
The proposed DQL	$\mathcal{O}\left(T_{epoch} \times I \times L \times n_{\text{action}}^2\right)$	4.05×10^{10}

$$\mathcal{O}\left(R_q^2\right) \times \mathcal{O}\left(\Upsilon^2 \log\left(\frac{1}{\epsilon}\right)\right) \times \left(\mathcal{O}\left(\Upsilon^2\right) + \mathcal{O}\left(\Upsilon R_q\right)\right) = \mathcal{O}\left(\Upsilon^3 \times R_q^2 \times \log\left(\frac{1}{\varepsilon}\right) \times \max\left\{\Upsilon, R_q\right\}\right). \quad (24)$$

4.3.2. DQL with the water-filling

In DQL, the number of neurons in the input layer is equal to the dimension of the state, and the number of neurons in the output layer is equal to the discrete size of the action space. Here, as shown in Fig. 1, we design DQL as a fully connected network consisting of L hidden layers and no bias with $(n_{\text{state}} = 4 + 2K)$ input neurons and $\left(n_{\text{action}} = A_q \times P_q \times E_q\right)$ output neurons. In this paper, the number of neurons in all layers of the hidden layer is set to be equal to the number of actions n_{action}; thus, the computation of the first input layer is $\mathcal{O}\left(n_{\text{state}} \times n_{\text{action}}\right) = \mathcal{O}\left(2K \times n_{\text{action}}\right)$. Let n_l be the number of neurons of the hidden layer l, where $1 \le l \le L-1$. Then, the time complexities for computing from the l-th hidden layer to $(l+1)$-th hidden layer, and from the L-th hidden layer to the output layer are $\mathcal{O}\left(n_{l-1}n_l + n_l n_{l+1}\right)$ and $\mathcal{O}\left(n_{L-1}n_L + n_L n_{\text{action}}\right)$, respectively Du et al. (2018). Because the number of neurons in the hidden layers for the DQL is equal to the number of neurons in the output layer, the time complexity in a feed-forward network can be calculated as $\sum_{i=1}^{L-1} \mathcal{O}\left(n_{\text{action}}^2\right) = \mathcal{O}\left((L-1) \times n_{\text{action}}^2\right)$. Suppose that the proposed algorithm converges with T_{epoch} epochs and I iterations. Therefore, the total number of samples in the simulation process is $T_{epoch} \times I$. It is important to note that the proposed algorithm performs the training offline and $n_{\text{action}} \gg 2K$. Therefore, the total time complexity of the proposed algorithm for deploying is

$$\begin{aligned}&\mathcal{O}\left(T_{epoch} \times I\right) \times \left[\mathcal{O}\left((L-1)\, n_{\text{action}}^2\right) + \mathcal{O}\left(2K n_{\text{action}}\right)\right] \\ &= \mathcal{O}\left(T_{epoch} \times I \times n_{\text{action}} \max\left\{L \times n_{\text{action}}, 2K\right\}\right) \\ &= \mathcal{O}\left(T_{epoch} \times I \times L \times n_{\text{action}}^2\right). \end{aligned} \quad (25)$$

Table 1 summarizes the comparisons of the computational complexities of the time complexity analysis of the successive hover-and-fly and DQL with the water-filling algorithm. For better understanding, we provide an example to evaluate the computational complexity of these algorithms when $R_q = 10^4$, $\varepsilon = 10^{-5}$, $T_{epoch} = 10$, $I = 10^3$, $L = 5$, $A_q = 9$, $P_q = 10$, and $E_q = 10$. The result shows that the DQL with the water-filling algorithm can significantly reduce computational complexity when compared to the successive hover-and-fly algorithm.

5. Performance evaluation

In this section, we evaluate the performance of the proposed DQL with the water-filling algorithm in the context of the UAV-enabled WPT network with array antennas to transmit energy to the GUs. These are randomly deployed over the coverage area with random initial battery capacity determined using the uniform distribution. In our simulation, the UAV is equipped with 8×8 array antennas, which are divided into four 4×4 sub-array antennas [8]. The departure location of the UAV is set as the center of the coverage area (R, R), and flight altitude is fixed at a certain altitude $H = 10$ m following the effectiveness of the WPT, [31].

Table 2
Environment setup.

Parameters	Value
Cell radius R	50 m
Number of GUs K	10
Total flying time T	1200 s
Time slot duration δ	1 s
UAV's maximum speed V	10 m/s
UAV's altitude H	10 m
Pathloss exponent η	3.6
Rician fading gain	5 dB m
Energy conversion factor η	50%
Number of array antenna	8×8
Array antenna block	4×4
GU's maximum battery	1 W h
UAV's maximum battery E_{UAV}	$\{1, 2, ..., 10\}$ kW h
Elevation angle maximum	45 degree
Quantize x and y-axis M	1000

Table 3
Simulation parameters for the proposed DRL.

Parameters	Value
Learning rate α	0.01
Number of hidden layer L	5 layers
ϵ-greedy ϵ	0.1
Discount factor γ	0.99
Replay memory size D	100
Optimizer	SGD

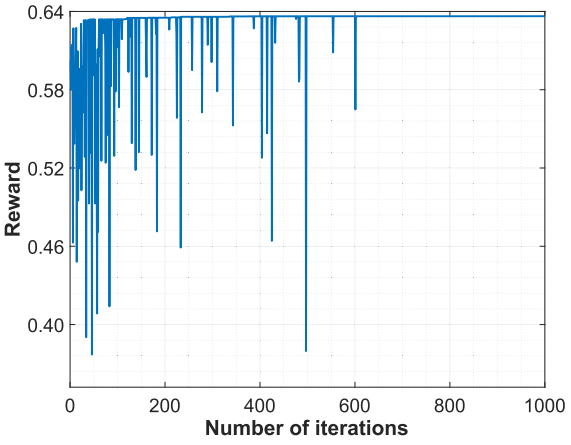
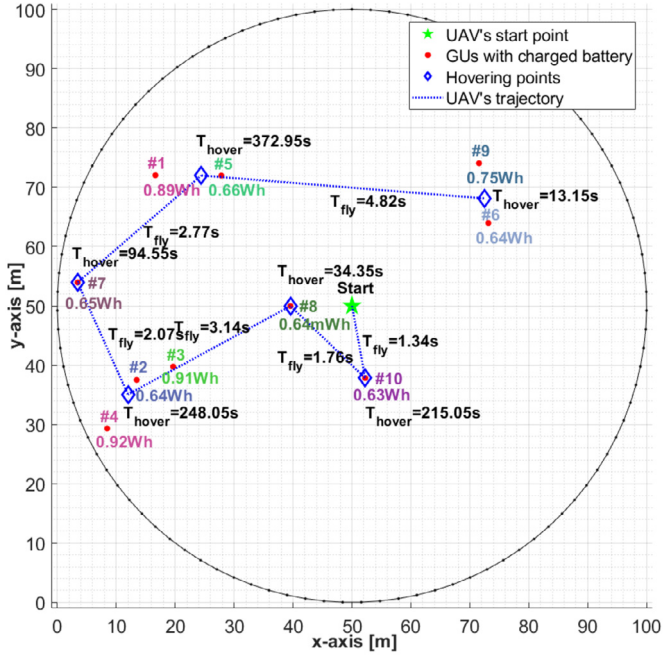


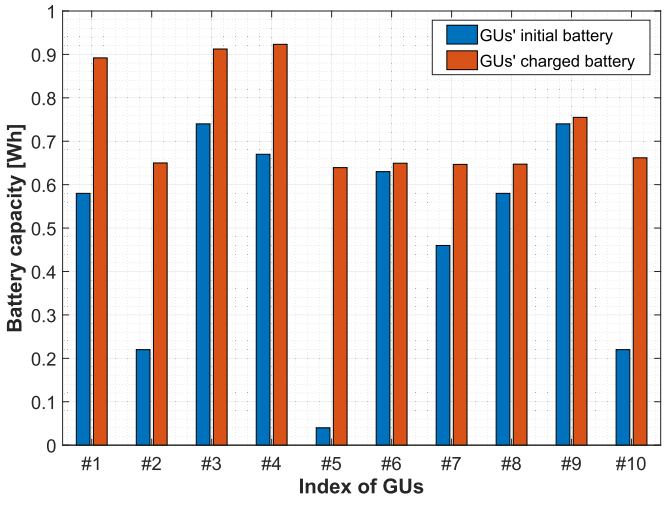
Fig. 4. Training convergence of proposed DRL.

Moreover, in this study, we compare our proposed algorithm with the successive hover-and-fly algorithm and the static hovering scheme. In the static hovering scheme, the UAV hovers on top of each GU and moves from one hovering point to another instantly, where the hovering duration is calculated based on the water-filling algorithm in Algorithm 1. Table 2 provides a summary of the simulation settings that are utilized to set up the environment. It is noted that the total flying time in each simulation run is fixed as 1200 seconds, which is suitable amount of time for the UAV to effectively perform the proposed and comparison algorithms, and evaluate their performances [12]. We use stochastic gradient descent (SGD) in the proposed DQL with the water-filling approach, and the parameters linked to the DRL model are given in Table 3. To perform the simulation of the proposed method, we used MATLAB 2020b with Deep Learning Toolbox.

In order to demonstrate the training convergence, we generate 10 GUs with randomly distributed initial battery levels, set the maximum battery capacity of the UAV to 6 kW h, and fix the maximum transmit power to 34 dB m. Fig. 4 illustrates the training of the proposed DRL, which converges after approximately 600



(a) UAV's trajectory with flying and hovering duration.

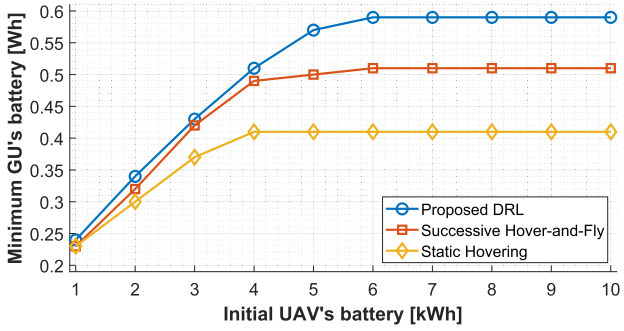


(b) GUs' initial vs. charged battery.

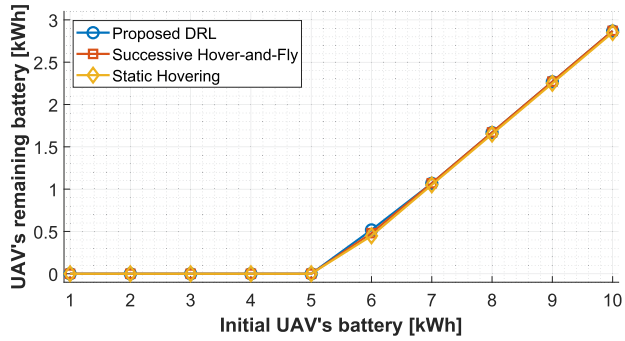
Fig. 5. Simulation result of the proposed DRL with 10 GUs.

rounds of iterations, assuming that the network environment remains constant. It is noted that the graph initially fluctuates because it aims for exploration, but after 600 iterations, it stabilizes for exploitation and the deep Q-network (DQN) is optimized.

Fig. 5(a) shows the generated environment and the UAV's trajectory with flying and hovering duration for the finite time duration $T = 1200$ s and the maximum UAV transmission power $P_{\max} = 30$ dB m. In this environment, the UAV finds the trajectory and hovering duration to transmit the energy for the GUs. The UAV hovers longer at the fifth hovering point than at other hovering points because our reward design gives the highest priority to hovering at the lowest-charged battery, GU #5 with 0.04 W h. It is noted that the energy transmitted by the UAV to GUs is dependent not only on the position of the GUs but also on the minimum battery level of GUs. In Fig. 5(b), the bar chart shows the battery charges of GUs at the initial stage and after being charged by the



(a) Minimum GUs' battery vs. initial UAV's battery with $T = 1200s$.



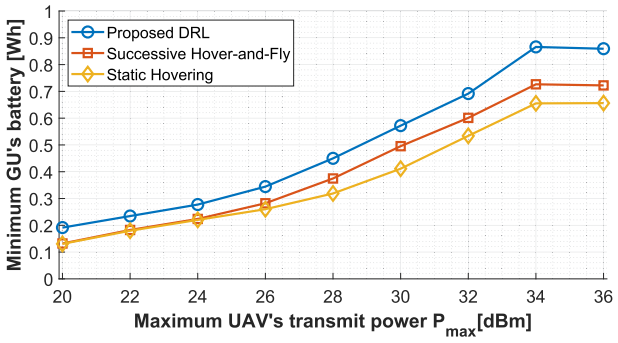
(b) UAV's remaining battery vs. initial UAV's battery with $T = 1200s$.

Fig. 6. The simulation result for different initial UAV's batteries.

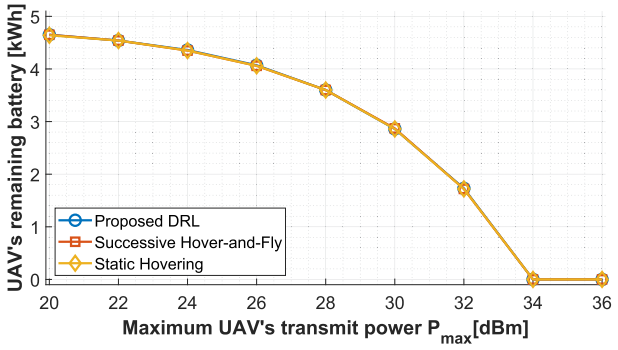
UAV. As seen in this figure, the proposed method maximizes the lowest GU battery charge, where GUs with the battery levels have received almost the same amount of energy from the UAV. From the result, we can verify that our proposed DQL with the water-filling algorithm can effectively solve the problem of maximizing the lowest GU energy by jointly optimizing the UAV's trajectory, beamforming pattern, and transmit power.

Fig. 6, we compare the effectiveness of the proposed method with the successive hover-and-fly trajectory algorithm proposed in [27]. The successive hover-and-fly trajectory algorithm achieves the result using an exhaustive 2D search and solving the traveling salesman problem (TSP). Fig. 6(a) shows the minimum GU battery level after being charged by a UAV against the initial UAV battery level with constant flying-time duration $T = 1200$ s and UAV's transmit power $P_{\max} = 30$ dB m. The graph shows that the proposed DRL method outperforms the successive hover-and-fly trajectory algorithm and static hovering scheme. This graph also shows that, even if the initial UAV's battery capacity is increased, the minimum GU's battery capacity does not increase because the UAV's time flying period and maximum transmitting power are fixed. Fig. 6(b) shows the remaining battery of the UAV after charging the GUs by using each method. The result shows that the remaining battery of the UAV is almost the same in each case. It is noted that when we have a low initial battery capacity on the UAV, the remaining UAV will run out of battery because the UAV delivers all its energy to the GUs during the flying period. Otherwise, although we have an initially high battery level on the UAV, it does not increase the minimum GU battery life because of the fixed flying time constraint.

Fig. 7(a) shows the minimum GU battery level versus the maximum transmit power $P_{\max}$ of the UAV for a constant flying time duration $T = 1200$ s and UAV initial battery $E_{\max} = 10$ kW h. The



(a) Minimum GUs' battery vs. maximum UAV's transmit power with $T = 1200s$.



(b) UAV's remaining battery vs. maximum UAV's transmit power with $T = 1200s$.

Fig. 7. The simulation result for different UAV's transmit powers.

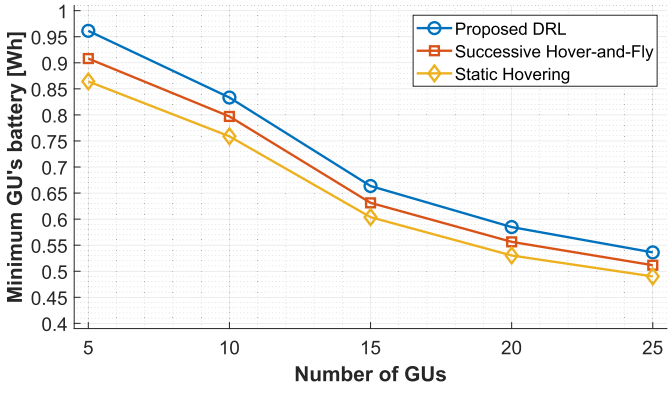


Fig. 8. Minimum GUs' battery vs. number of GUs.

result shows that the minimum GU battery level increases exponentially for $P_{\max} \in \{20, 22, ..., 34\}$ dB m; otherwise, it remains stable after $P_{\max} = 34$ dB m. It is clear that the result will remain stable, although we increase the UAV's transmit power because of the limit of the UAV's initial battery. The graph also shows that our proposed method outperforms the successive hover-and-fly trajectory algorithm and static hovering scheme. And Fig. 7(b) shows that the proposed method and successive hover-and-fly algorithm achieve the same performance in terms of the remaining battery level of the UAV.

Furthermore, we evaluate the performance of our proposed DRL algorithm with varying number of GUs (5, 10, 15, 20, and 25) and the comparison is conducted under the same conditions, as shown in Fig. 8. In this figure, the initial battery of the UAV remains at 6 kW h, and the maximum transmit power is fixed at 32 dB m. The result shows that the minimum battery level of the GUs decreases as the number of GUs increases due to the UAV's battery limitation, and our proposed DRL algorithm outperforms the consecutive hover-and-fly algorithm irrespective of the given number of GUs.

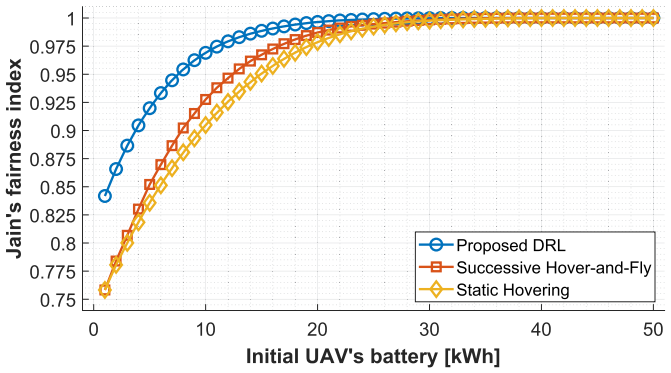


Fig. 9. Jain's fairness of GUs' battery.

In Fig. 9, we evaluate and compare the fairness of the algorithms using Jain's fairness index under the same condition where the number of GUs is set to 25, the initial battery of the UAV is set to 6 kW h, and the maximum transmit power is fixed by 32 dB m. Here, Jain's fairness index [16] is calculated as

$$J(E_1, E_2, ..., E_K) = \frac{\left(\sum_{k=1}^{K} E_k\right)^2}{n \times \sum_{k=1}^{K} E_k^2}, \tag{26}$$

where E_k is the battery level of the GU-k charged by the UAV. The graph indicates that increasing the initial battery of the UAV leads to an increase in Jain's fairness index to one. It is clear that increasing initial battery level of the UAV results in charging all GUs until their batteries are full, ensuring fair distribution of energy among them. Additionally, the result demonstrates that the proposed DRL algorithm outperforms the other schemes.

6. Conclusions

This paper studied jointly optimizing the UAV's trajectory, beamforming pattern, and transmit power to maximize the lowest GU energy, in which the UAV is adapted with the array antennas for delivering wireless energy to charge the GUs. To solve this problem, we first converted the problem into a discrete-time format and then transformed it again into a grid world problem. After that, we proposed a DQL design with the water-filling algorithm to find the optimal UAV trajectory, transmit power, and beamforming pattern. Simulation results reveal that the proposed algorithm significantly enhances the lowest GU energy received compared to the successive hover-and-fly algorithm with low-time computation.

On the other hand, the altitude of the UAV is an important parameter which determines the effect of the algorithm in the context of WPT network. And using multi-UAV in this network scenario is an effective method to tackle network scalability issue. Therefore, in our future work, we will explore a multi-agent DRL approach assuming multi-UAV scenario and build a joint optimization model where the altitude of a UAV is one of optimization parameters.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] M. Alzenad, A. El-Keyi, F. Lagum, H. Yanikomeroglu, 3-d placement of an unmanned aerial vehicle base station (uav-bs) for energy-efficient maximal coverage, IEEE Wirel. Commun. Lett. 6 (2017) 434–437.

[2] P. Angeletti, M. Lisi, Multimode beamforming networks for space applications, IEEE Antennas Propag. Mag. 56 (2014) 62–78.

[3] P.S. Bithas, E.T. Michailidis, N. Nomikos, D. Vouyioukas, A.G. Kanatas, A survey on machine-learning techniques for uav-based communications, Sensors 19 (2019) 5170.

[4] S. Bubeck, et al., Convex optimization: algorithms and complexity, Found. Trends Mach. Learn. 8 (2015) 231–357.

[5] H. Chen, G. Wang, Z. Wang, H.-C. So, H.V. Poor, Non-line-of-sight node localization based on semi-definite programming in wireless sensor networks, IEEE Trans. Wirel. Commun. 11 (2011) 108–116.

[6] P. Dayan, C. Watkins, Q-learning, Mach. Learn. 8 (1992) 279–292.

[7] Y. Du, K. Yang, K. Wang, G. Zhang, Y. Zhao, D. Chen, Joint resources and workflow scheduling in uav-enabled wirelessly-powered mec for iot systems, IEEE Trans. Veh. Technol. 68 (2019) 10187–10200.

[8] W. Feng, N. Zhao, S. Ao, J. Tang, X. Zhang, Y. Fu, D.K. So, K.-K. Wong, Joint 3d trajectory design and time allocation for uav-enabled wireless power transfer networks, IEEE Trans. Veh. Technol. 69 (2020) 9265–9278.

[9] G. Gui, H. Huang, Y. Song, H. Sari, Deep learning for an effective nonorthogonal multiple access scheme, IEEE Trans. Veh. Technol. 67 (2018) 8440–8450.

[10] G. Gui, F. Liu, J. Sun, J. Yang, Z. Zhou, D. Zhao, Flight delay prediction based on aviation big data and machine learning, IEEE Trans. Veh. Technol. 69 (2019) 140–150.

[11] S. Hayat, E. Yanmaz, R. Muzaffar, Survey on unmanned aerial vehicle networks for civil applications: a communications viewpoint, IEEE Commun. Surv. Tutor. 18 (2016) 2624–2661.

[12] X. He, J. Bito, M.M. Tentzeris, A drone-based wireless power transfer anc communications platform, in: 2017 IEEE Wireless Power Transfer Conference (WPTC), IEEE, 2017, pp. 1–4.

[13] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari, F. Adachi, Deep learning for physical-layer 5g wireless techniques: opportunities, challenges and solutions, IEEE Wirel. Commun. 27 (2019) 214–222.

[14] H. Huang, Y. Peng, J. Yang, W. Xia, G. Gui, Fast beamforming design via deep learning, IEEE Trans. Veh. Technol. 69 (2019) 1065–1069.

[15] R. Hunger, Floating Point Operations in Matrix-Vector Calculus, vol. 2019, Munich University of Technology, Inst. for Circuit Theory and Signal, 2005.

[16] R.K. Jain, D.-M.W. Chiu, W.R. Hawe, et al., A quantitative measure of fairness and discrimination, Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA, 21, 1984.

[17] L. Liu, K. Xiong, J. Cao, Y. Lu, P. Fan, K.B. Letaief, Average aoi minimization in uav-assisted data collection with rf wireless power transfer: a deep reinforcement learning scheme, IEEE Int. Things J. 9 (2021) 5216–5228.

[18] A. Massa, G. Oliveri, F. Viani, P. Rocca, Array designs for long-distance wireless power transmission: state-of-the-art and innovative solutions, Proc. IEEE 101 (2013) 1464–1481.

[19] F.S. Melo, Convergence of q-learning: a simple proof, Tech. Rep., Institute of Systems and Robotics, 2001, pp. 1–4.

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing Atari with deep reinforcement learning, arXiv preprint, arXiv:1312.5602, 2013.

[21] M. Nariman, F. Shirinfar, A.P. Toda, S. Pamarti, A. Rofougaran, F. De Flaviis, A compact 60-GHz wireless power transfer system, IEEE Trans. Microw. Theory Tech. 64 (2016) 2664–2677.

[22] K.K. Nguyen, A. Masaracchia, V. Sharma, H.V. Poor, T.Q. Duong, Ris-assisted uav communications for iot with wireless power transfer using deep reinforcement learning, IEEE J. Sel. Top. Signal Process. 16 (2022) 1086–1096.

[23] F. Ono, H. Ochiai, R. Miura, A wireless relay network based on unmanned aircraft system with rate optimization, IEEE Trans. Wirel. Commun. 15 (2016) 7699–7708, https://doi.org/10.1109/TWC.2016.2606388.

[24] A.M. Seid, J. Lu, H.N. Abishu, T.A. Ayall, Blockchain-enabled task offloading with energy harvesting in multi-uav-assisted iot networks: a multi-agent drl approach, IEEE J. Sel. Areas Commun. 40 (2022) 3517–3532.

[25] Q. Sun, Y. Zhao, H. Du, B. Yu, M. Gao, X. Jin, A multi-drone joint map building method based on multi-variable layer directed cross genetic algorithm, in: 2022 41st Chinese Control Conference (CCC), 2022, pp. 6788–6792.

[26] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.

[27] L. Xie, J. Xu, R. Zhang, Throughput maximization for uav-enabled wireless powered communication networks, IEEE Int. Things J. 6 (2018) 1690–1703.

[28] J. Xu, Y. Zeng, R. Zhang, Uav-enabled wireless power transfer: trajectory design and energy optimization, IEEE Trans. Wirel. Commun. 17 (2018) 5092–5106.

[29] H. Yan, Y. Chen, S.-H. Yang, Uav-enabled wireless power transfer with base station charging and uav power consumption, IEEE Trans. Veh. Technol. 69 (2020) 12883–12896, https://doi.org/10.1109/TVT.2020.3015246.

[30] Z. Yang, C. Pan, M. Shikh-Bahaei, W. Xu, M. Chen, M. Elkashlan, A. Nallanathan, Joint altitude, beamwidth, location, and bandwidth optimization for uav-enabled communications, IEEE Commun. Lett. 22 (2018) 1716–1719.

[31] X. Yuan, Y. Hu, A. Schmeink, Joint design of uav trajectory and directional antenna orientation in uav-enabled wireless power transfer networks, IEEE J. Sel. Areas Commun. 39 (2021) 3081–3096.

[32] X. Yuan, T. Yang, Y. Hu, J. Xu, A. Schmeink, Trajectory design for uav-enabled multiuser wireless power transfer with nonlinear energy harvesting, IEEE Trans. Wirel. Commun. 20 (2020) 1105–1121.

[33] Y. Zeng, X. Xu, R. Zhang, Trajectory design for completion time minimization in uav-enabled multicasting, IEEE Trans. Wirel. Commun. 17 (2018) 2233–2246.

[34] Y. Zeng, R. Zhang, Energy-efficient uav communication with trajectory optimization, IEEE Trans. Wirel. Commun. 16 (2017) 3747–3760.

[35] Z. Zhang, H. Pang, A. Georgiadis, C. Cecati, Wireless power transfer-an overview, IEEE Trans. Ind. Electron. 66 (2018) 1044–1058.

[36] Z. Zhang, X. Xie, C. Xu, R. Wu, Energy harvesting-based uav-assisted vehicular edge computing: a deep reinforcement learning approach, in: 2022 IEEE/CIC International Conference on Communications in China (ICCC Workshops), IEEE, 2022, pp. 199–204.

[37] L. Zhu, J. Zhang, Z. Xiao, X. Cao, D.O. Wu, X.-G. Xia, 3-d beamforming for flexible coverage in millimeter-wave uav communications, IEEE Wirel. Commun. Lett. 8 (2019) 837–840.