Age of Information-Aware Trajectory Optimization for Time-Sensitive UAV Systems in Uplink SCMA Networks

Teshager Hailemariam Moges[a], Thanh Phung Truong[b], Demeke Shumeye Lakew[c], Thien Ho Huong[d], Vinh Truong Hoang[d], Nhu-Ngoc Dao[a], Sungrae Cho[b]

[a]*Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea*
[b]*School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea*
[c]*Department of Computer Science, Kombolcha Institute of Technology, Wollo University, Dessie 1145, Ethiopia*
[d]*Faculty of Computer Science, Ho Chi Minh City Open University, Ho Chi Minh City 70000, Vietnam*

Abstract

Optimizing the Age of Information (AoI) and minimizing mission time are pivotal in uncrewed aerial vehicle (UAV)-assisted Internet of Things (IoT) systems to maintain data freshness for time-sensitive applications, particularly in dynamic mobile environments. This paper tackles these objectives in sparse code multiple access (SCMA) networks, a potential access technology for sixth-generation IoT communications. In particular, we optimized UAV trajectory by jointly considering AoI and mission time reductions along with time-critical constraints. Accounting for the dynamic and evolving interactions between IoT devices and UAVs, we modeled the optimization problem as a Markov decision process and resolved it by a deep deterministic policy gradient-based dynamic hovering point selection algorithm (DDPG-DHPSA). Our approach especially manages current AoIs and data volumes to dynamically determine the optimal hovering point sequence forming the UAV trajectory. Simulation results demonstrated that our solution reduces AoI and mission time by approximately 87% and 73%, respectively, outperforming recent benchmarks.

Keywords: Age of information, data collection, sparse code multiple access, uncrewed aerial vehicle

1. Introduction

Internet of Things (IoT) has emerged as a crucial enabler in an era dominated by artificial intelligence, where the demand for smart solutions across sectors is ever-increasing [1]. Advancements in artificial intelligence and networking have created safeguarding technology to connect humans and integrate objects, collectively known as IoT devices (IoTDs), to realize the vision of global digitization. These devices are integral to the evolving landscape of fifth- and sixth-generation (6G) technology. Integrating uncrewed aerial vehicles (UAVs) with the IoT is a promising solution because UAVs effectively collect data in IoT ecosystems. Their ability to offer flexible and cost-effective wireless communication and computational services is a significant advantage, allowing them to hover over IoTDs and gather data [2]. Despite these benefits, UAVs have limitations, such as low transmit power, limited battery life, and compact antenna size, affecting data processing [3].

An effective communication design is critical in UAV-assisted IoT systems, particularly under dynamic and challenging network conditions in modern mobile networks. In this context, sparse code multiple access (SCMA) emerges as a promising access scheme for IoT communications in 6G networks, owing to its advantages in enhancing spectral efficiency and enabling massive connectivity. As a special non-orthogonal multiple access (NOMA) approach, SCMA employs multidimensional constellations and sparse codewords to support overlapping transmissions [4]. By utilizing these multidimensional sparse codewords, SCMA markedly boosts spectral efficiency and connectivity, facilitating simultaneous transmissions with minimized interference [5]. In contrast to orthogonal multiple access techniques, SCMA allows concurrent access without strict spectrum partitioning, leading to improved bandwidth utilization. Moreover, compared to conventional NOMA methods, SCMA streamlines multi-user detection with efficient message-passing algorithms, lowering decoding complexity and latency [6]. These unique features make SCMA particularly appropriate for real-time, high-density IoT applications, such as UAV-assisted networks with stringent and dynamic AoI demands. Nevertheless, the critical challenges of minimizing the age of information (AoI) and mission time remain in such dynamic network environments. An effective strategy for addressing these challenges involves optimizing the UAV trajectory, encompassing three central components: dynamically clustering IoTDs, identifying optimal hovering point (HP) positions, and determining the sequence of UAV visits.

To address this problem, we proposed a deep deterministic policy gradient (DDPG)-based dynamic HP selection algorithm (DHPSA) to optimize the HPs of the UAV trajectory to collect IoT data. The selection of optimal HPs is based on the AoI and data volume of the IoTD clusters, prioritizing those with a higher

Email addresses: 22110612@sju.ac.kr (Teshager Hailemariam Moges), tptruong@uclab.re.kr (Thanh Phung Truong), demeke@uclab.re.kr (Demeke Shumeye Lakew), thien.hh@ou.edu.vn (Thien Ho Huong), vinh.th@ou.edu.vn (Vinh Truong Hoang), nndao@sejong.ac.kr (Nhu-Ngoc Dao), srcho@cau.ac.kr (Sungrae Cho)

AoI and data volume. In summary, the main contributions of this paper are noted below.

- We consider time-sensitive communication between the UAV and IoTDs in the uplink SCMA. We formulate the UAV trajectory optimization problem as a mixed-integer nonconvex problem to minimize the weighted sum of the AoI of the IoTDs and the UAV mission time in clusters simultaneously. This problem is subject to constraints, including the UAV's first and last three-dimensional location, dynamic IoTD locations, and AoI.
- To address the complexity of this nondeterministic polynomial-time-hard problem and capture the dynamic nature of the environment, we transform the original optimization problem into a Markov decision process (MDP). Then, to address the problem, we apply dynamic k-means clustering to categorize IoTDs based on their proximity and determine the optimal number of clusters using the elbow method. We propose the *DDPG-DHPSA* for selecting optimal HPs in every cluster. This algorithm intelligently considers two critical metrics: the higher AoI and larger data volume of IoTDs. By prioritizing these factors, the DDPG-DHPSA aims to minimize the mission time and AoI in this dynamic UAV-IoT environment.
- Finally, we validate the performance of the proposed algorithm by conducting simulations and comparing them with several baseline schemes. The results reveal that the proposed algorithm achieves significant improvement in terms of mission time and AoI minimization in various scenarios.

The remainder of this paper is organized as follows. Section 2 presents the related work. Next, Section 3 defines the system model and problem formulation. Then, Section 4 introduces the *DDPG-DHPSA*. Section 5 provides the simulation results, performance evaluations, and discussion. Finally, Section 6 concludes the paper.

2. Related Work

The UAV is a strong candidate for collecting data from IoTDs due to its flexibility, mobility, and ability to cover large areas [7]. Thus, UAVs are promoted and applied to improve data collection focused on optimizing UAV trajectories [8]. In particular, Fu *et al.* investigated the application of UAV to collect data in both maritime and ground IoT system scenarios [9, 10, 11]. The findings in these studies expose the potential of UAVs to assist various mobile services when UAV trajectory and communication are optimally designed. Zhu *et al.* [8] used UAVs to investigate their promising role in data collection to minimize the total mission time by optimizing the UAV trajectory. Several studies [12, 13, 14, 15] have addressed various contexts to minimize the AoI, including in UAV-assisted networks. The expected weighted sum of the AoI was minimized [12] by optimizing the UAV trajectory and scheduling policies by employing the DDPG algorithm. Liu *et al.* [13] minimized the average AoI by optimizing the UAV trajectory, data transfer scheduling, and energy harvesting of ground sensors using the DQN algorithm. Sun *et al.* [14] jointly optimized the UAV flight speed and HP positions using deep reinforcement learning (DRL) to minimize the weighted expected AoI. Zhang *et al.* [15] minimized the average AoI by optimizing the trajectory of the UAV using the twin delayed DDPG algorithm.

Likewise, mission time is a critical objective in UAV-based data collection. Reducing mission time plays a role in applications where real-time data are critical, such as traffic monitoring, and allows for more frequent data collection. Several studies have focused on minimizing the mission time [16, 17, 18, 19]. Zhu *et al.* [16] jointly optimized the UAV trajectory and wake-up time allocation to minimize the mission competition time in a data collection scenario using SCA. Employing SCA to optimize the UAV trajectory minimizes the completion time [17]. The mission completion time was minimized using the min–max multiple traveling salesperson problem algorithm in [18]. Gao *et al.* [19] adopted the SCA approach to minimize the maximum mission completion time by refining the mission allocation, trajectory, and speed. Although algorithms, such as SCA and the genetic algorithm, are employed in optimizing the UAV trajectory, DRL has become a robust approach to solving complex and continuous optimization problems, including UAV trajectory optimization [20]. Recent studies [21, 22, 23] proposed by Chapnevis *et al.* jointly optimized UAV trajectory for AoI and mission time reduction in various scenarios with and without infrastructures. The problem was formulated to be reasonable by Integer Linear Programming (ILP) methods.

In contrast, Liu *et al.* [24] studied the UAV trajectory planning problem to reduce the UAV mission completion time by jointly optimizing the UAV flying speeds, HP positions, and visiting sequences. They decomposed the problem into UAV speed optimization and UAV path optimization problems. They employed the SCA and genetic algorithm to solve these problems. Then, the authors integrated the solutions into an AoI-and-energy-aware trajectory optimization algorithm to solve the primary optimization problem. Although they used the AoI as a critical constraint, there are significant differences between their work and the proposed method. A joint optimization of clustering, transmission, and trajectory design has been proposed in [25] to address the AoI minimization problem in UAV-assisted wireless-powered communication networks. The authors introduced a clustering-based dynamic adjustment of the shortest path algorithm to minimize long-term average AoI. This approach demonstrated significant performance improvements by dynamically adjusting UAV trajectories based on real-time energy constraints and data freshness. Despite its strengths, this work did not consider SCMA protocol advantages, which we explicitly address in our study.

Despite the extensive research on these individual optimization objectives, either the AoI or mission time, few studies have taken a holistic approach to addressing UAV trajectory optimization considering the dual objectives of minimizing the AoI and mission time in modern mobile networks, especially in an uplink SCMA environment.

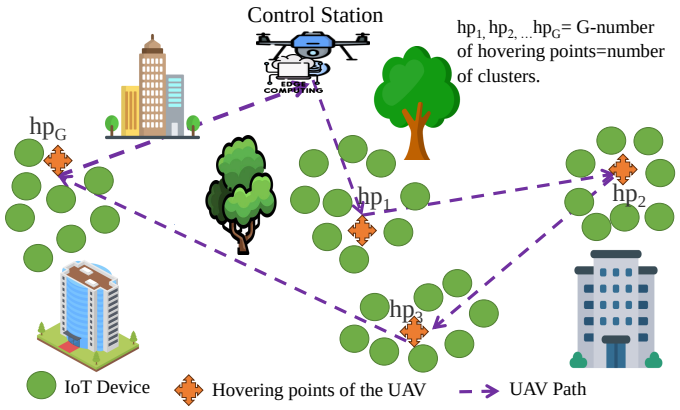


Figure 1: SCMA-based UAV-assisted data collection model.

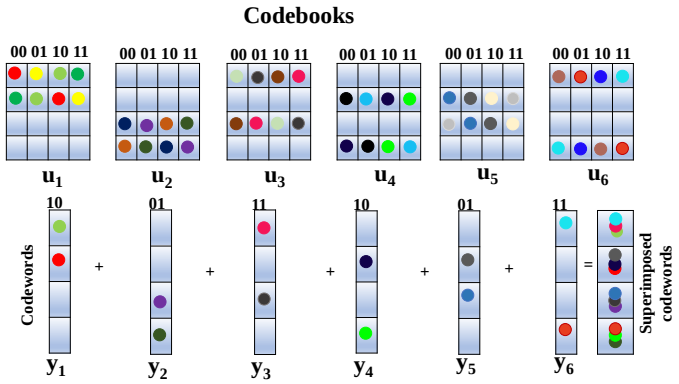


Figure 2: Sparse code multiple access (SCMA) codebooks encoding and multiplexing an SCMA uplink system.

3. System Model and Problem Formulation

A framework consisting of a single UAV and N IoTDs with G clusters is considered, as illustrated in Fig. 1, where the UAV visits all clusters. Table 1 summarizes the notation in this paper. We propose an SCMA-based UAV-assisted IoT network comprising a control station s_0 and multiple IoTDs. The IoTDs (the users) are denoted by $\mathbb{U} = \{1, 2, \ldots, N\}$, where N represents the total number of IoTDs. These devices are spatially and unevenly distributed in a defined area and are dynamically grouped into clusters to facilitate efficient data management and collection.

The set of clusters at any given time t can be defined as $C(t) = \{c_1, c_2, \ldots, c_G\}$, where G denotes the total number of clusters, as depicted in Fig. 1. Each IoTD u belongs to exactly one cluster and is represented by the subset $\mathcal{U}_{c_i} \subseteq \mathbb{U}$. Then, the position of the u^{th} IoTD in a cluster c_i at time t, is defined as $w_u^{c_i}(t) = (x_u^{c_i}(t), y_u^{c_i}(t), 0)$. The initial location of the UAV is defined as $s(0) = (X(0), Y(0), Z(h))$. After it departs from the control station, the UAV position at time t is formulated as $\mathbf{p}(t) = (X(t), Y(t), Z(t))$, where each position $\mathbf{p}(t)$ depends on its previous position $\mathbf{p}(t-1)$. The complete trajectory of the UAV L includes the initial position, the dynamically determined optimal HPs in each visited cluster, and the final position back at the control station. These optimal HPs, identified by the proposed algorithm, are denoted by $\mathbf{H} = \{hp_1, hp_2, \ldots, hp_G\}$, where each HP hp_i is associated with a specific cluster c_i. The HP position at time t in a target cluster is defined as $\mathbf{u}^{c_i}(t) = (X^{c_i}(t), Y^{c_i}(t), Z^{c_i}(t))$. The UAV flight path should start and end at $s(0)$ to maximize the data collection and guarantee the safe return of the UAV to the control station.

The UAV hovers over the selected HPs in each cluster to collect data from the IoTDs. The IoTDs send report messages, including timestamps (to verify the generation time of IoTD data), data volumes, and their locations, to the UAV. Upon reaching a cluster, the IoTD activates high-speed, long-range new radio links to transmit the data to the UAV, in which 3GPP Rel. 17 supports communication standards for cellular IoT [26].

3.1. K-means Dynamic Clustering Model

Clustering plays a pivotal role in UAV trajectories by reducing the travel distance and communication overhead and providing efficient path planning. The k-means model is a widely used clustering algorithm that partitions a dataset into k clusters based on the similarity of data points [27]. Moreover, the IoTDs are mobile; thus, dynamic k-means clustering is used, which is illustrated in this paper. We introduced a binary indicator δ_{w_u,c_i} to represent the link between IoTDs and their allocated clusters. This link indicates whether a specific IoTD w_u belongs to cluster c_i and is defined as follows:

$$\delta_{w_u,c_i} = \begin{cases} 1, & \text{if IoTD } w_u \text{ is in cluster } c_i, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The following constraint ensures that each IoTD is assigned to only one cluster:

$$\sum_{c_i \in C} \delta_{w_u,c_i} = 1, \quad \forall w_u \in \mathcal{U}, \tag{2}$$

where C represents the set of all clusters, and $\mathcal{U}$ denotes the set of all IoTDs.

The set of clusters C_{vis} visited by a UAV is tracked as follows:

$$C_{vis} = \{c_i \in C \mid \sigma_{c_i} = 1\}, \tag{3}$$

where σ_{c_i} denotes a binary indicator that verifies whether the UAV has visited cluster c_i, which is critical for optimizing UAV flight routes.

3.2. SCMA Communication Model

This paper considers a mission-critical network where the UAV communicates with IoTDs via SCMA technology. The SCMA channel serves as the uplink channel access method for simultaneous task offloading from multiple IoTDs to the UAV. Moreover, SCMA is distinguished by its distinct codebooks and orthogonal resources (subcarriers), facilitating interference-free communication.

The set of IoTDs in cluster c_i is denoted by $\mathbb{U}_{c_i} = \{1, 2, \ldots, n\}$ as where n is the total number of IoTDs in that cluster. The total number of orthogonal resources (subcarriers) is defined as $\mathcal{J} = \{1, 2, \ldots, J\}$, and the total number of available SCMA codebooks for the IoTDs is denoted as $\mathcal{K} = \{1, 2, \ldots, K\}$. The allocation matrix at time t, $\mathbf{F}_{\text{SCMA}}(t) \in \mathcal{K}^{U \times K}$, determines the distribution of the subcarriers among the SCMA codebooks

Table 1: Notation definitions

Symbol	Definition	Symbol	Definition
N	Total number of IoTDs in the system	G	Total number of clusters
$k_u(t)$	SCMA codebook assigned to IoTD u at time t	u	Index of an individual IoTD in a cluster ($u = 1, 2, ..., U$)
$w_u^{c_i}[t]$	Position of IoTD u at time t	$p[t]$	UAV hovering point position at time t
β_0	Path loss at the reference distance	α	Path-loss exponent
N_0	Noise power	p_u	Transmit power of IoTD u on subcarrier j
B	Bandwidth	$d_{cr}^{c_i}(t)$	Euclidean distance from the current UAV location to the target cluster c_i at time t
K	Set of SCMA codebooks	$\psi_{u,k_u}(t)$	Indicator function for codebook assignment at time t
$\nu_{j,k_u}(t)$	Proportion of power assigned to resource j within codebook k_u at time t	$\text{AoI}_{\max}$	Maximum allowable AoI for device u in the network
$\text{AoI}_u^{c_i}$	AoI of device u in a cluster c_i	AoI^{c_i}	AoI in cluster c_i
$\text{DTT}(w_u^{c_i})$	Data transmission time in cluster c_i	$T_{cp,u}^{c_i}$	Data computation time for IoTD u in cluster c_i
$T_{tr}^{c_i}$	Transit time to a cluster	$T_{mission}$	Total mission time of a UAV

allocated to the IoTDs.

$$\mathbf{F}_{\text{SCMA}}(t) = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}. \tag{4}$$

How SCMA works Each IoTD u encodes a set of information bits into a codeword to prepare for transmission. This process involves converting $U = \log_2 F$ information bits, represented by vector $I_u(t) = [I_{k,1}, I_{k,2}, \ldots, I_{k,U}]^T$, into a complex codeword for transmission. The chosen codeword, denoted by $x_u = [x_{u,1}, x_{u,2}, \ldots, x_{u,M}]^T$, is selected from the SCMA codebook k_u assigned to the IoTD. The dimensions of the SCMA codebooks are represented by $M \times F$, where M specifies the codeword dimensionality, and F represents the number of codewords in each codebook. The selection of sparse codewords ($d_r < M$) from these codebooks enables effective decoding by the UAV, even when signals from multiple IoTDs overlap. In the classical example of the SCMA configuration of $U = 6$ and $J = 4$, each variable node connects to d_r function nodes, and each function node is linked to d_f variable nodes, demonstrating the connectivity and resource-sharing dynamics.

Each IoTD u in cluster c_i is assigned a unique SCMA codebook k_u from a set of K codebooks, ensuring orthogonal communication channels and minimizing interference among IoTDs. The allocation matrix $\mathbf{F}_{\text{SCMA}}(t)$, with elements ($f_{j,k_u}$, indicates whether the j^{th} subcarrier is allocated to IoTD u via codebook k_u at time t. If $f_{j,k_u} = 1$, then codebook k_u occupies the j^{th} subcarrier as indicated in the matrix equation in (4). The SCMA framework allows for communication channel overloading, which occurs when the number of IoTDs U exceeds the number of accessible subcarriers J. The framework is represented by an overloading factor $\lambda = \frac{U}{J} > 1$. Due to the sparsity of SCMA codewords, multiple IoTDs can share the same subcarrier with minimal interference.

The primary link between each IoTD in cluster c_i and the UAV is assumed to follow a line-of-sight path, which is typical in UAV communication scenarios due to their elevated positions. For each time instant t, the channel power gain h_{uav} between the UAV and an IoTD u in cluster c_i can be accurately modeled as follows:

$$h_{uav,u}(t) = \beta_0 d_{cr,c_i}^{-\alpha}(t) = \frac{\beta_0}{\left(Z^2 + \left|\mathbf{p_u}(t) - \mathbf{w}_u^{c_i}(t)\right|^2\right)^{\frac{\alpha}{2}}}, \tag{5}$$

where β_0 denotes the average channel power gain at a reference distance $d_0 = 1$ m. In addition, $d_{cr,c_i}^{-\alpha}(t)$ represents the distance between IoTD u in cluster c_i and the UAV, accounting for the altitude Z and the horizontal distance between the position $\mathbf{p}(t)$ of the UAV and the position $\mathbf{w}_u^{c_i}(t)$ of the IoTD. Further, α denotes the path-loss exponent, indicating the rate at which the signal power decreases with distance.

Each IoTD u is assigned a unique SCMA codebook k_u from a set of K codebooks for the efficient use of spectral resources and to minimize interference, ensuring orthogonal communication channels. This assignment is indicated by $\psi_{u,k_u}(t)$, where $\psi_{u,k_u}(t) = 1$ if codebook k_u is assigned to IoTD u at time t, to minimize interference among IoTDs; otherwise, $\psi_{u,k_u}(t) = 0$. An IoTD u is assumed to occupy only one SCMA codebook to reduce interference (i.e., $\sum_{k_u \in K} \psi_{u,k_u}(t) = 1$). Moreover, p_u represents the transmit power of the device assigned to resource j in proportion to $\nu_{j,k_u}(t) \in [0, 1]$, ensuring the total power does not exceed the IoTD power budget of $\sum_{j \in k_u} \nu_{j,k_u}(t) = 1$.

Interference is expected when multiple IoTDs communicate simultaneously. The interference of IoTD u on the j^{th} resource is defined as follows:

$$I_{j,u}(t) = \sum_{i \in \{|h_{j,i}(t)|^2 > |h_{j,u}(t)|^2\}} p_i \nu_{j,k_u}(t) |h_{j,i}(t)|^2. \tag{6}$$

All IoTDs are set to a fixed transmit power $p_u(t)$ and B, the available bandwidth. The signal-to-interference-plus-noise ratio for IoTD u using subcarrier j at time t is defined as follows:

$$\gamma_{j,u}(t) = \frac{\psi_{u,k_u}(t) \nu_{j,k_u}(t) p_u(t) |h_{j,u}(t)|^2}{N_0 + I_{j,u}(t)} \tag{7}$$

where $|h_{j,u}(t)|^2$ represents the channel gain, N_0 is the noise power, and $I_{j,u}(t)$ captures the interference from other IoTDs sharing subcarrier j.

The transmit rate $R_{j,u}(t)$ for IoTD u on subcarrier j at time t is computed using the signal-to-interference-plus-noise ratio:

$$R_{j,u}^{c_i}(t) = \log_2(1 + \gamma_{j,u}(t)). \tag{8}$$

Thus, the overall transmit rate for IoTD u at time t, considering all allocated subcarriers, is given by

$$R_u^{c_i}(t) = \sum_{j \in \mathcal{J}} \psi_{u,k_u}(t) R_{j,u}(t). \tag{9}$$

3.3. Age of Information

The AoI is a critical parameter that assesses the freshness of data, which is vital for mission-critical applications, such as healthcare and telemedicine, traffic management and control, and disaster management. For each IoTD u in cluster c_i, the AoI at timeslot t, denoted $\text{AoI}_u^{c_i}$, is the time elapsed since the most recent update from u was successfully received and processed by the UAV. When the UAV arrives at a selected cluster, individual IoTDs with a higher AoI are prioritized for data updates to guarantee efficient management of information freshness in the cluster. Furthermore, an AoI threshold, $\text{AoI}_{\max}$, is established to maintain the relevance and timeliness of the information across the network as follows:

$$\text{AoI}_u^{c_i} \leq \text{AoI}_{\max} \quad \forall u \in \mathcal{U}. \tag{10}$$

The AoI of cluster c_i at timeslot t, $\text{AoI}^{c_i}(t)$, is defined as the maximum AoI observed among all IoTDs within that cluster. Hence,

$$\text{AoI}^{c_i}(t) = \max_{\forall u \in c_i} \{\text{AoI}_u^{c_i}\} \leq \text{AoI}_{\max}. \tag{11}$$

This definition reflects the worst-case freshness of information within the cluster, ensuring that the cluster's AoI captures the most outdated data point among its constituent devices.

3.4. Time Consumption Model

In the proposed scenario, the mission time in this system constitutes the transit/flight time, data transmission time (DTT), and data computation time.

3.4.1. UAV Transit Time

The UAV flight time or transit time ($T_{tr}^{c_i}$) model computes the time for the UAV to travel to each cluster from the control station. For a given cluster c_i, the transit time is calculated based on the distance from the current location to the HP in a cluster divided by the constant speed v of the UAV, calculated as follows:

$$T_{tr}^{c_i} = \frac{d_{cr,c_i}}{v}, \tag{12}$$

where d_{cr,c_i} is the Euclidean distance from the current location of the UAV to cluster c_i.

3.4.2. Data Transmission Time

In addition to the flight time, the DTT ($\text{DTT}^{c_i}u$) is another critical component, referring to the time it takes for data to be transmitted from the IoTDs to a UAV. The time depends on the data volume Vu^{c_i} and the upload rate $R_u^{c_i}$:

$$\text{DTT}(_u^{c_i}) = \frac{Vu^{c_i}}{R_u^{c_i}}. \tag{13}$$

The total transmission time for cluster c_i is the sum of the transmission time for each IoTD in a cluster:

$$T_{DTT}^{c_i} = \sum_{u=1}^{U} \text{DTT}(_u^{c_i}). \tag{14}$$

3.4.3. Data Computation Time

The UAV is assumed to be equipped with a low edge-computing capacity to process the data from IoTDs partially while collecting it. If all computational tasks from IoTDs in cluster c_i are offloaded to the UAV, the computational time for each IoTD u, denoted as $T_{cp,u}^{c_i}$, depends on the computational workload $c_{r,u}^{c_i}$ and the assigned computing resource f_c. Thus, the computational time is given by the following:

$$T_{cp,u}^{c_i} = \frac{c_{r,u}^{c_i}}{f_c}. \tag{15}$$

Then, the cumulative computational time for cluster c_i is the sum of the computational time for all IoTDs in the cluster:

$$T_{cp}^{c_i} = \sum_{u=1}^{U} T_{cp,u}^{c_i}. \tag{16}$$

3.4.4. Clusterwise Time Consumption

The total time spent on the UAV mission in each cluster c_i is defined as T^{c_i}. The total UAV mission time in a cluster is the sum of the transit time between clusters ($T_{tr}^{c_i}$) and the maximum of the total DTT ($T_{DTT}^{c_i}$) and computational time ($T_{cp}^{c_i}$). The max function is used because the UAV collects and processes data at the same time, enabling taking the maximum time among the transmission and computation time. With this logic, the time the UAV spends in each cluster for transmission and processing is defined as follows:

$$T^{c_i} = T_{tr}^{c_i} + \max(T_{DTT}^{c_i} + T_{cp}^{c_i}). \tag{17}$$

3.5. Optimization Problem

The optimization problem is given by:

$$\begin{aligned} \min_{\boldsymbol{H}} \quad & \sum_{i=1}^{G} (\alpha T^{c_i}(t) + (1-\alpha) \cdot \text{AoI}^{c_i}(t)) \\ \text{s.t.} \quad & (11), \\ & \text{C1: } \sum_{c \in C} \delta_{u,c_i} = 1 \quad \forall u \in \mathcal{U}, \\ & \text{C2: } Z_{\min} \leq Z_u^{c_i}(t) \leq Z_{\max}, \\ & \text{C3: } \sigma_{c,u} \leq 1 \quad \forall c \in C. \end{aligned} \tag{18}$$

where α represents a weighting factor between the mission time and AoI. Constraint **C1** forces every IoTD to join only one cluster, while **C2** manages the heights of the UAV between the defined minimum and maximum altitude levels. Further, **C3** ensures that the UAV visits a cluster only once.

The formulated optimization problem presents a nonconvex optimization problem, incorporating discrete variables related to clustering decisions, UAV hovering points, and continuous variables representing UAV trajectory coordinates and timing constraints. This complexity arises due to the inherent nonlinearity and coupling of trajectory, communication, and computation decisions, leading to high-dimensional search spaces. Furthermore, the presence of real-time AoI constraints and the dynamic mobility of IoT devices make it much more challenging to find a solution, requiring advanced heuristic or reinforcement learning-based approaches for practical and efficient solutions.

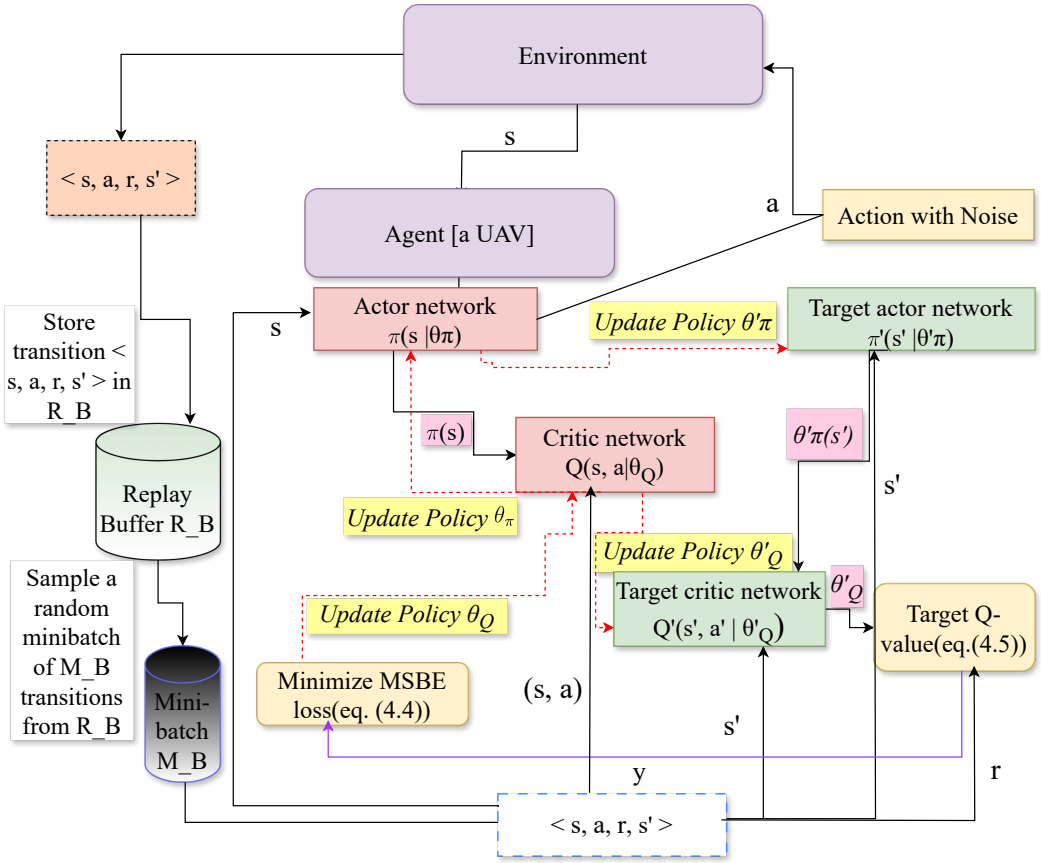


Figure 3: The proposed *DDPG-DHPSA* model.

4. Proposed DDPG-DHPSA Algorithm

Fortunately, DRL has recently emerged as an efficient method for tackling nonconvex problems in communication networks [28, 29]. Nevertheless, clustering IoTDs and finding an optimal trajectory remain significant challenges, given the vast number of possible combinations. Accordingly, we propose a DRL–based framework that combines clustering, deep reinforcement learning, and heuristic trajectory-finding to address these tasks. Although numerous DRL algorithms could theoretically be substituted, DDPG has demonstrated particular effectiveness and stability in similar settings [30, 31]. For this reason, this study's primary goal is to integrate clustering with DRL and heuristic path planning to solve the problem.

This section provides a DRL technique called *DDPG–DHPSA*, which is based on DDPG, to solve the optimization problem. First, this section discusses the primer of DDPG and why DDPG is best suited for solving the problem. Then, the proposed problem is transformed as an MDP framework. Finally, this section demonstrates how to apply the *DDPG–DHPSA* to the proposed model.

4.1. DDPG Primer

The DDPG model, an excellent model-free algorithm in the sphere of DRL, is suitable for high-dimensional continuous spaces [32]. This approach combines value-based and policy-based methods, making it ideal for optimizing the UAV trajectory. In the actor network, weights (θ_π) are periodically updated, current actions a' are selected based on the perceived state s', and s' and R are calculated by interacting with the environment, defined as $\pi, (a = \pi(s|\theta_\pi)$. In response to the current environmental state, the actor selects an action carefully chosen from a range of available actions. The critic network updates the weight θ_Q of the value network and provides the current Q value, $Q = Q(s, a|\theta_Q)$. The critic network estimates the value of the state-action pairs, guiding the decisions of the actor. Then, the policy updates for the actor evaluate its actions, considering the present environmental state, and send the feedback to the policy network for improvement. Using the Bellman equation, the optimal accumulative Q-value of the critic is defined as follows:

$$Q_*(s, a|\theta_Q) = \max_{a\in\mathcal{A}, r, s'\sim E}\left[r(s, a) + \gamma \max_{a'} Q_*(s', a'|\theta_Q)\right], \quad (19)$$

where r represents the reward from the environment following the execution of action a in state s, $\gamma \in [0, 1]$ denotes the discount factor that reduces the significance of subsequent rewards, and s' and a' indicate the action to be performed and the subsequent state.

The DDPG framework combines the best policy determination and Q-learning techniques. The critical objective of implementing this framework is for the policy to deterministically link states to their optimal actions to maximize the expected cumulative reward, denoted by $J(s|\theta_\pi)$. The expected cumulative reward is derived based on state s. According to Q-learning

principles, the learning process of the critic relies on the Bellman equation (19). Based on the policy gradient approach, the parameters of the actor are adjusted as follows:

$$\begin{aligned}\nabla_{\theta_\pi} J(\theta_\pi) &= \mathbb{E}\left[\nabla_{\theta_\pi} Q\left(s, a|\theta_Q\right)\right] \\ &= \mathbb{E}\left[\nabla_a Q\left(s, a|\theta_Q\right) \nabla_{\theta_\pi}\left(a\right)\right] \\ &= \mathbb{E}\left[\nabla_a Q\left(s, a|\theta_Q\right) \nabla_{\theta_\pi}\left(\pi'(s)\right)\right]. \end{aligned} \tag{20}$$

Moreover, the DDPG algorithm enhances the original actor policy $\pi(s|\theta_\pi)$ to encourage exploration by adding noise $\mathcal{N}$ generated by the Ornstein–Uhlenbeck process:

$$\alpha_n = \pi(s|\theta_\pi) + \mathcal{N}. \tag{21}$$

The target actor and target critic networks stabilize the learning process. Using uniform sampling from the replay buffer, the actor target network θ'_π determines action a' based on state s'. The critic target network θ'_Q calculates the current value Q and derives the target value based on it.

The replay buffer $\mathcal{R}_\mathcal{B}$ stores experience tuples (s, a, r, s'). The minibatches $\mathcal{M}_\mathcal{B}$ are sampled from this buffer to update the value and policy networks. Then, the mean squared Bellman error loss is computed from the minibatches. Thus, for the critic network, the loss function $L(\theta_Q)$ is calculated as the mean squared error between the predicted Q-values and the target values as follows:

$$L(\theta_Q) = \mathbb{E}_{s,a,r,s' \sim \mathcal{M}_\mathcal{B}}\left[\left(Q(s, a|\theta_Q) - y\right)^2\right]. \tag{22}$$

In this case, $Q(s, a|\theta_Q)$ is the Q-value predicted by the critic network for state s and action a with parameter θ_Q. The expectation $\mathbb{E}$ is taken over the minibatch of experiences (s, a, r, s') sampled from the replay buffer $\mathcal{M}_\mathcal{B}$. Further, y is the target value derived from the optimal cumulative $Q_*(s, a|\theta_Q)$). Based on the Bellman equation (19), the target value y is computed as follows:

$$y = r(s, a) + \gamma Q'_{\theta'_Q}\left(s', \pi'_{\theta'_\pi}(s')\right). \tag{23}$$

As defined in (22), the objective is to minimize the mean squared Bellman error loss to ensure the Q-function ($Q(s, a|\theta_Q)$) aligns closely with the target value y. To maintain stability while minimizing the mean squared Bellman error, the parameters of both the target actor network ($\theta'\pi$) and the target critic network ($\theta' Q$) are updated during each iteration of the primary network update cycle with τ as a target network update rate, as follows:

$$\begin{aligned}\theta'_\pi &\leftarrow \tau\theta_\pi + (1-\tau)\theta'_\pi \\ \theta'_Q &\leftarrow \tau\theta_Q + (1-\tau)\theta'_Q. \end{aligned} \tag{24}$$

4.2. Markov Decision Process Formulation

In the proposed system, the UAV operates as the DRL agent, making decisions based on its interactions with the dynamic IoT environment. The role of the UAV involves gathering and analyzing network data, akin to a central decision-making unit that manages and optimizes the information flow. Hence, the UAV trajectory optimization problem is formulated as an MDP, defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, including a state space $\mathcal{S}$, an action space $\mathcal{A}$, a reward function ($r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$), a state transition p, and a discount factor $\gamma \in [0, 1]$.

Algorithm 1 Dynamic clustering

Input: Set of IoT devices $\mathcal{U}$ (with $|\mathcal{U}| = N$), initial number of clusters G_{init}, maximum iterations max_{iter}, convergence threshold ϵ, quality threshold q_{thresh}, minimum clusters G_{min}, and maximum clusters G_{max}
Output: Cluster assignments $\{C_1, \ldots, C_G\}$

$G \leftarrow G_{init}$
Initialize centroids $\{c_1, \ldots, c_G\}$ using optimal HPs $\mathbf{H}$
$t \leftarrow 1$
loop
  repeat
    Assign each $u_j \in \mathcal{U}$ to the nearest centroid c_i
    Update centroids c_i to the mean of the assigned points
    $\Delta \leftarrow \max_i \|c'_i - c_i\|_2$
    $iter \leftarrow iter + 1$
  until $\Delta < \epsilon$ **or** $iter = max_{iter}$
  $\Delta_{env} \leftarrow \sum_{u \in \mathcal{U}} \|u_t - u_{t-1}\|_2 + |\mathcal{U}_t| - |\mathcal{U}_{t-1}|$
  if $\Delta_{env} > q_{thresh}$ **then**
    Initialize the $WCSS$ array
    for $g = G_{min}$ to G_{max} **do**
      Run the k-means algorithm with g clusters on $\mathcal{U}$, starting from the previous centroids
      Calculate $WCSS[g - G_{min}]$
    end for
    $G_{new} \leftarrow$ ElbowMethod($WCSS$)
    if $G_{new} \neq G$ **then**
      if $G_{new} > G$ **then**
        Add $G_{new} - G$ centroids near the current centroids
      else
        Remove $G - G_{new}$ centroids with the least data points
      end if
      $G \leftarrow G_{new}$
      $C \leftarrow$ InitializeClusters($\mathcal{U}, G$)
    end if
  end if
  $t \leftarrow t + 1$
end loop
return C

State Space: The state space $\mathcal{S}$ in the SCMA-based UAV-assisted IoT network comprises the location of IoTDs and a UAV, the data volume, and the AoI of each device. Each state $s \in \mathcal{S}$ is defined as a vector:

$$s_t = \langle \mathbb{P}^{c_i}(t), \mathbb{W}_u^{c_i}(t), \mathbb{A}_u^{c_i}(t), \mathbb{V}_u^{c_i}(t) \rangle, \tag{25}$$

where $\mathbb{P}^{c_i}(t) = (X^{c_i}(t), Y^{c_i}(t), Z^{c_i}(t))$ and $\mathbb{W}_u^{c_i}(t) = (x_u^{c_i}(t), y_u^{c_i}(t), 0)$ denote the location of the UAV and IoTDs, respectively. In addition, $(\mathbb{A}_u^{c_i}(t)$ represents a vector of the AoI for each IoTD in the cluster, and $\mathbb{V}_u^{c_i}(t)$ denotes the vector of the pending data volumes for each IoTD in the cluster at time t.

Action Space: By adjusting its strategy to HP positioning in the chosen cluster for optimal data collection, *DDPG-DHPSA* determines the following action of the UAV based on the observed state. At every point in its trajectory, $a \in \mathcal{A}$ represents a decision the UAV makes, allowing it to dynamically alter its flight path and data collection strategy in response to the changing environment at time t:

$$a_t = \langle H_u^{c_i}(t) \rangle, \tag{26}$$

where $H_u^{c_i}(t)$ denotes the HP in the selected cluster, defined as $H_u^{c_i}(t) = (X_u^{c_i}(t), Y_u^{c_i}(t), Z_u^{c_i}(t)$.

State Transition Probability: The state transition probability acts as a probabilistic indicator for transitioning from one state $s \in \mathbb{S}$ to another state $s' \in \mathbb{S}$ after executing action $a \in \mathbb{A}$ in a given system. This probability quantifies the likelihood of

such state changes (with probability $P(s'|s,a)$. The state transition probability considering the Markov property is defined as follows:

$$P(s'|s,a) = \mathbb{P}^{c_i}(s'), \mathbb{W}_u^{c_i}(S'), \mathbb{A}_u^{c_i}(s'), \mathbb{V}_u^{c_i}(s') \\ |\mathbb{P}^{c_i}(s), \mathbb{W}_u^{c_i}(s), \mathbb{A}_u^{c_i}(s), \mathbb{V}_u^{c_i}(s)). \quad (27)$$

In practice, the environment's dynamics can yield numerous possible next states, arising from combinations of $\mathbb{P}^{c_i}(t)$, $\mathbb{W}_u^{c_i}(t)$, $\mathbb{A}_u^{c_i}(t)$, and $\mathbb{V}_u^{c_i}(t)$. By sampling the environment, the transition probability, $P(s'_t \mid s_t, a_t)$, can be estimated for each particular next state $s'_t \in \mathcal{S}'$, with $\mathcal{S}'$ denoting the set of all possible next states reachable from s_t under action a_t. Through extensive training with a large number of samples, the model learns these transition probabilities and captures the environment's behavior, thus guiding the selection of actions that optimizes system performance over the long term.

Reward Function: The reward function $r : S \times A \rightarrow \mathbb{R}$ encourages actions that reduce the overall flight time and AoI while ensuring efficient data collection from IoTDs. The effectiveness of achieving the objectives of the UAV actions is quantified. Actions that reduce AoI and increase the data collection time are rewarded, whereas actions that increase AoI and prolong the mission duration are penalized. The agent is responsible for identifying an optimal policy π_* that maximizes the expected reward. Hence, the Q-value is described as follows:

$$Q(s,a|\theta_Q) = \mathbb{E}\left[\sum_{t=1}^{T} \gamma^{t-1}(r|s,a)\right], \quad (28)$$

where T denotes the number of time slots, and the optimal policy is defined as follows:

$$\pi_* = \arg\max_{\pi} Q(s,a|\theta_Q), \quad \forall s \in \mathcal{S}. \quad (29)$$

The reward r is defined as the negative of the weighted sum of the time and AoI in a clusterwise manner:

$$r_t(s_t,a_t) = -\sum_{i=1}^{G} \left(\alpha T^{c_i}[t] + (1-\alpha)\cdot AoI^{c_i}[t] + \xi + P\right), \quad (30)$$

where ξ denotes a coverage bonus reward to encourage the UAV to cover all devices in the entire cluster, the UAV obtains a relatively large penalty P when it does not meet the constraints set in (18). Moreover, P comprises penalties for critical conditions P_c and operational constraints P_o. The penalty associated with critical conditions, P_c, is calculated as $P_c = \gamma_c \cdot n_c$, where γ_c is the penalty coefficient per device that exceeds the critical thresholds for AoI, and n_c represents the number of such devices. The operational constraints P_o are penalized by $P_o = \gamma_o$ if violated and by 0 otherwise. Thus, the overall penalty P is given by $P = P_c + P_o$.

4.3. Our Solution

In our dynamic k-means clustering approach, IoT devices are grouped based on their real-time spatial positions. Initially, we determine a predefined number of clusters using the elbow method, which involves analyzing the within-cluster sum of squares (WCSS) for different cluster counts. The optimal number of clusters is selected at the "elbow" point where further improvements in WCSS become marginal. Then, clusters are updated dynamically at regular intervals or when a significant displacement is detected to address the mobility of IoT devices. During each update, devices are reassigned to their nearest cluster centroids. These centroids are then recalculated iteratively until convergence is reached or a maximum iteration limit is exceeded, ensuring that the clustering process remains both efficient and adaptive to changes in mobility patterns. As described in Algorithm 1, the algorithm starts by initializing key variables: the initial number of clusters (G_{init}), the centroids, and the UAV's starting position ($s(0)$). At each time step, IoT devices are assigned to their nearest centroid, and the centroids are updated until the convergence criterion—defined by a threshold, ε. Accordingly, the UAV's position is updated using a normalized random direction vector while ensuring that altitude constraints ($Z_{\min}$ and $Z_{\max}$) are satisfied. Additionally, environmental changes are monitored by assessing the movements of both the IoT devices and the UAV. If these movements exceed a specified threshold δ, the algorithm re-evaluates the number of clusters. This is done by running the k-means algorithm for various cluster counts (from $G_{\min}$ to $G_{\max}$) and recalculating the WCSS. The optimal number of clusters is then determined using the elbow method, and the centroids are adjusted accordingly.

The *DDPG-DHPSA* is strategically designed to minimize the overall mission time and AoI. For each cluster C_i, the DDPG algorithm employs reinforcement learning to determine the optimal HP H_i based on the AoI and data volume of the IoTDs in the cluster. The optimized HPs for all clusters are collected into the set $H = hp_1, hp_2, \ldots, hp_G$. Algorithm 2 provides the detailed pseudocode. In addition, Fig. 3 presents the architecture and workflow of the *DDPG-DHPSA*. It is worthy noting that the algorithm is regularly recalculated each time the UAV departs from its current cluster to move to the next hovering location.

4.4. Complexity Analysis

The framework comprises two main components: dynamic clustering and the DDPG algorithm for determining the hovering points. Therefore, the total complexity of the proposed framework is the sum of these components.

According to [33], the complexity of the K-Means algorithm is given by

$$O_K = O(N \cdot K \cdot D \cdot I), \quad (31)$$

where K, D, and I denote the number of clusters, the data dimensionality, and the number of iterations until convergence, respectively. As the environment scales, the complexity of the dynamic K-Means algorithm can be approximated by $O_K \approx O(N)$.

The time complexity of the DDPG algorithm is determined by the actor network architecture [34], expressed as

$$O_D = O\left(\sum_{l=1}^{L} \varsigma_{l-1}\,\varsigma_l\right), \quad (32)$$

where ς_l is the number of nodes in the l-th layer, and $L-1$ is the number of hidden layers. From (25) and (26), the input

Algorithm 2 *DDPG-DHPSA*

Input: The environment parameters $\mathbb{P}^{c_i}[t]$, $\mathbb{W}_u^{c_i}[t]$, $\mathbb{A}_u^{c_i}[t]$, and $\mathbb{V}_u^{c_i}[t]$ and the *DDPG-DHPSA* training parameters E, T_s, $\mathcal{R}_\mathcal{B}$, $\mathcal{M}_\mathcal{B}$, α_{actor}, α_{critic}, and γ, τ, α_n

Output: The trained weights of θ of the actor networks for the agent and optimized HPs $H = hp_1, hp_2, \ldots, hp_G$

Initialize the weights θ_π and θ_Q of the deep neural actor network $\pi(s)$ and critic network $Q(s)$, respectively
Initialize the weights θ_π and θ_Q of the deep neural actor network $\pi(s)$ and critic network $Q(s)$, respectively
Initialize the target weight $\theta'_\pi \leftarrow \theta_\mu$ of the target actor π' and target weight $\theta'_Q \leftarrow \theta_Q$ of the target critic Q'
Initialize the replay buffer size R_B and minibatch $\mathcal{M}_\mathcal{B}$; initialize a Gaussian process with $mean = 0$ and $var = 2$
Initialize $H \leftarrow \emptyset$
for $episode \leftarrow 1$ to E **do**
    Initialize state s based on (25)
    while $t \leq T_s$ **do**
        Observe state s
        Actor generates action a according to (26)
        Compute reward r according to (30)
        Observe the next state s'
        Store the tuple (s, a, r, s') in buffer R_B
        Sample minibatch $\mathcal{M}_\mathcal{B}$ tuples from R_B to train the deep neural network
        Update θ_Q by minimizing the loss in (22)
        Update θ_π from the policy gradient in (20)
        Update the target networks based on (24)
        Increment t
    end while
    Extract and store the optimized HPs H_i from the actor network for each cluster
    $H \leftarrow H \cup H_i$
end for
return θ_π, H

layer ($l = 0$) has $4N$ nodes, whereas the output layer ($l = L$) has 3 nodes corresponding to the hovering point. Hence, as the environment expands, the complexity of the DDPG-based hovering point determination can be approximated by $O_D \approx O(N)$.

Summarizing, the total time complexity of the proposed framework becomes $O(N+N) \approx O(N)$. Obviously, the proposed framework is feasible with respect to the environment scale in terms of the number of IoTDs N. However, in a large-scale environment requiring many clusters, a multi-UAV solution based on our framework could be employed, thereby making it possible to scale to more extensive networks.

5. Simulation Results

This section includes the simulation environment settings, simulations, and numerical results demonstrating the effectiveness of the proposed algorithm.

5.1. Simulation Environment and Setup

The simulation employed an Nvidia GeForce GTX 1650 GPU and Intel(R) Core(TM) i5-9400 CPU. Python 3.10.13 (64-bit) and PyTorch 1.13.1 with CUDA 11.6 support and Gym 0.26.2 were employed to program the software environment. In these experiments, we carefully considered the hyperparameters and training settings and recorded the performance measures for the analysis and evaluation along with the training progress.

Table 2: Environmental Parameters

Parameter	Value
β_0	30 (dB)
α	2
N_0	-174 (dBm/Hz)
Bandwidth	1 MHz
p_u	20 dBm
V_u	0.5 ~ 1 Mbits
Data-generation timestamp	0.1 ~ 0.3 s

In the simulation, we assumed that the UAV collected data from 100 IoTDs in a 200×200 m area at a constant speed of 15 m/s, aligning with the optimal maneuver speed for this configuration. The UAV flies at altitudes of between 60 and 300 m. For more details, Table 2 lists other environmental parameters, which are summarized based on [13, 35].

We set two hidden neural network layer values with 512 and 256 nodes with batch and buffer sizes of 16 and 50 000, respectively, and set the soft update coefficient to 0.01 and the discount factor to 0.99. We compared the proposed algorithm with four algorithms.

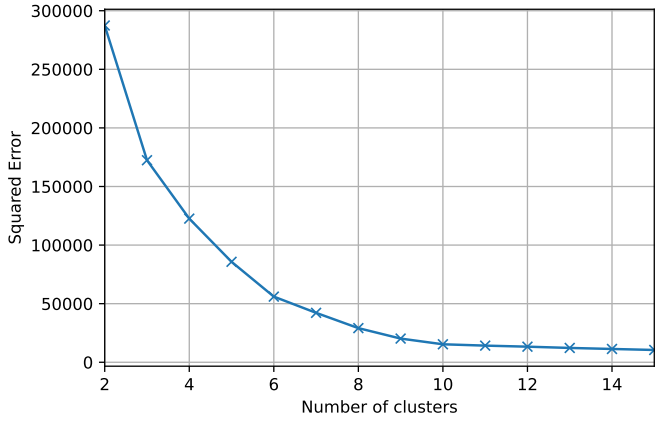
- **Proposed**: The proposed method is the main framework being evaluated, integrating three critical components: (1) a k-means dynamic algorithm to cluster IoTDs based on their spatial proximity, (2) a DDPG algorithm to optimize the HPs of the UAV in each cluster, and (3) an exhaustive search algorithm to find the optimal trajectory of the UAV across the clusters. The proposed framework combines these techniques to collect data from IoTDs while minimizing the AoI and mission time.

- **WoCluster**: The method without clustering (WoCluster) acts as a baseline comparison where IoTD clustering is not conducted. Instead, the DDPG method is implemented to identify the best HP while considering all IoTDs. The UAV moves to a single HP to gather data from all IoTDs. This approach provides an understanding of how clustering influences system performance.

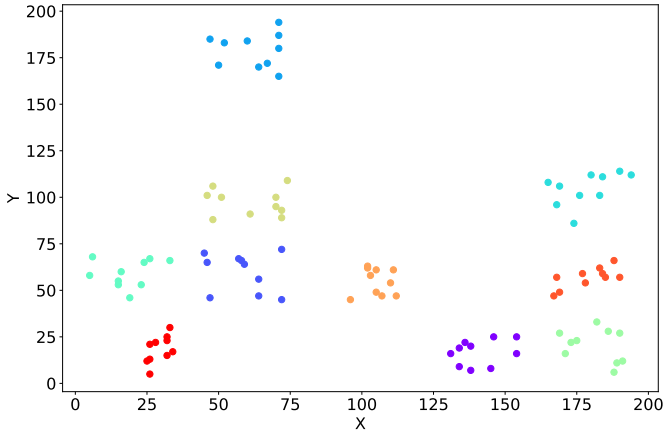
- **WoDDPG**: In the approach without DDPG (WoDDPG), HPs are established at the center of each cluster without optimization using the DDPG algorithm. The UAV visits each cluster center to collect data from the IoTDs.

- **PathGreedy**: The greedy technique described in [30, Sec. III-B] reduces the complexity of the exhaustive search for determining the trajectory of the UAV. The closest cluster to the present UAV location is the next cluster to be visited. Although this approach offers a computationally efficient method to plan trajectories in real time, a globally optimal solution cannot be guaranteed.

- **KRandom**: This approach selects the hovering locations and trajectory of the UAV at random after clustering the IoTDs using the k-means algorithm. This strategy serves as a baseline for comparing the framework performance in optimizing UAV positioning and path planning against a random method.



(a) K-means error by number of clusters



(b) Clustering results

Figure 4: Clustering Internet of Things devices (IoTDs) using the k-means algorithm.

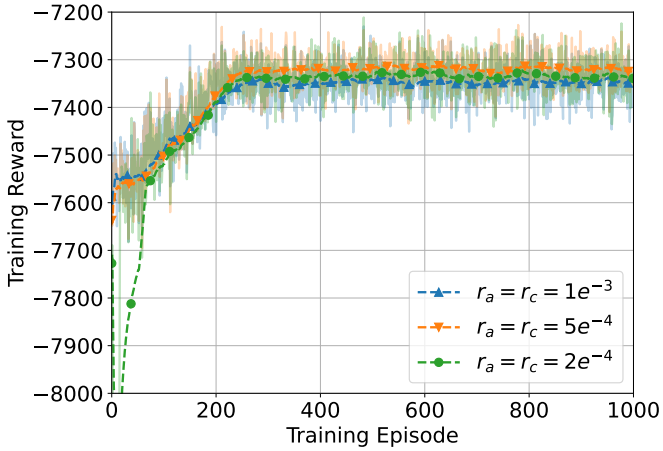


Figure 5: Convergence of the *DDPG-DHPSA*.

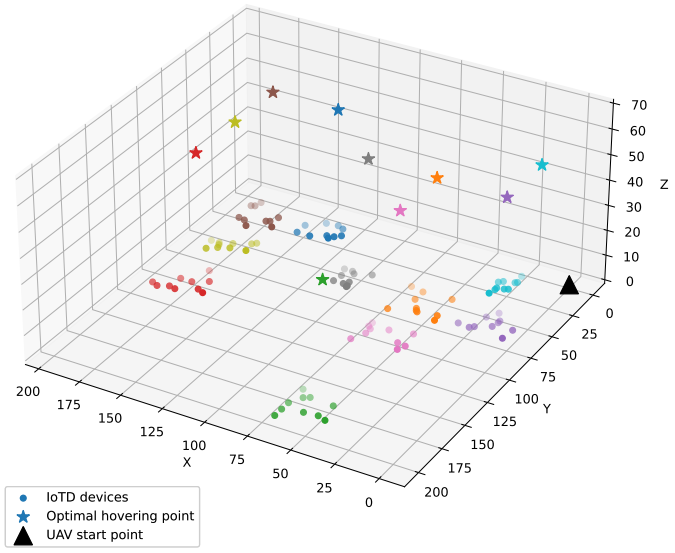


Figure 6: Hovering points according to Internet of Things device (IoTD) clusters.

5.2. *Proposed Framework Analysis*

First, we clustered the HPs by implementing the dynamic k-means algorithm. We applied continuous monitoring of the positions of the IoTDs, given that the devices are dynamic. When changes are detected, instead of reclustering the entire dataset, the algorithm assigns new IoTDs to the nearest cluster and updates the centroid, removing the removed IoTDs from their cluster and recalculating the centroid. Then, the algorithm reassigns the moved IoTDs if necessary and updates the affected centroids. To optimize the number of clusters in the data, we estimated the loss function, which was obtained by adding the squared errors between the cluster centroids and their data points for scenarios ranging from two to 15 clusters.

Fig. 4a presents the squared error decreases according to the increase in the number of clusters. The elbow method determined the optimal number of clusters, which was ten. Fig. 4b depicts the clustering results, with each cluster represented by distinct colors. This visualization effectively depicts the clustering structure and IoTD distribution among the clusters. Fig. 5 presents the training convergence of the algorithm under various learning-rate configurations. We explored various actor (r_a) and critic (r_c) learning rates, selecting values from $\{1e^{-3}, 5e^{-4}, 2e^{-4}\}$.

The training results reach convergence after about 250 episodes across all scenarios, and the case where $r_a = r_c = 5e^{-4}$ demonstrated the highest outcome. To optimize the HPs for each cluster, we employed the model trained in this best-case scenario. Thus, Fig. 6 depicts the optimal HPs of all clusters.

We applied the exhaustive search method, ensuring that all feasible cases were considered and explored to determine the most optimal trajectory. Fig. 7 displays the two-dimensional optimal trajectory.

5.3. *Performance Evaluation*

This section demonstrates the effectiveness of the proposed algorithm using extensive simulations.

Fig. 8 depicts the relationship between the data volume and AoI performance for the proposed algorithm and the WoDDPG, KRandom, WoCluster, and PathGreedy algorithms. The results demonstrate that the AoI scales linearly with increasing data volume across all five approaches. This proportional growth in the AoI can be attributed to the fact that larger data volumes require more time to transmit, leading to older information being received at the destination. When examining the proposed

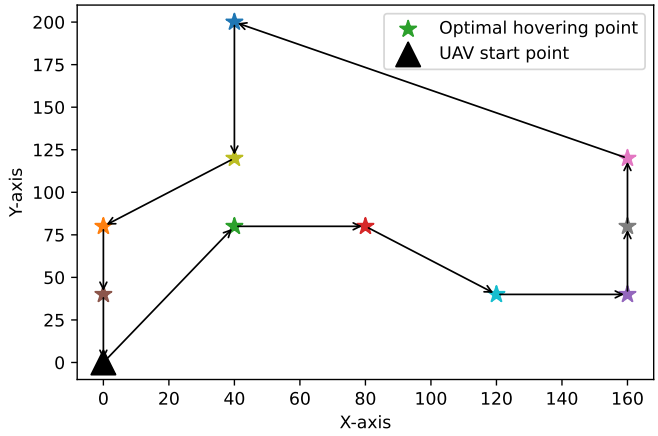


Figure 7: Optimal trajectory of the uncrewed aerial vehicle (UAV).

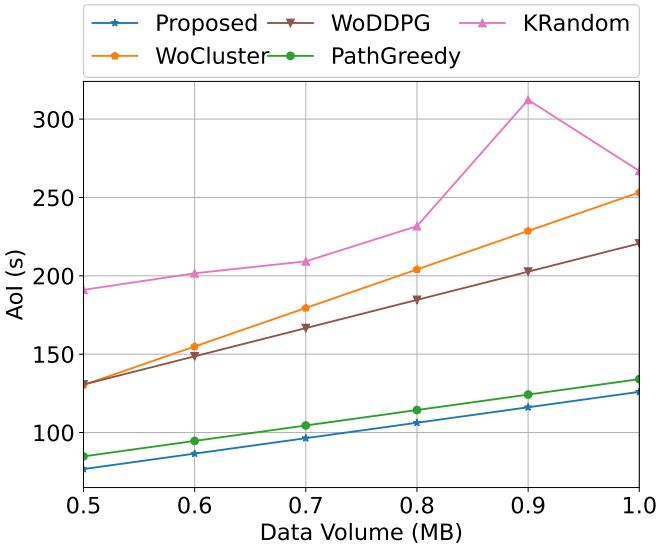


Figure 8: Performance by data volume.

algorithm, the AoI displays an 11% relative increase as the data volume increases by 0.1 MB. This substantial degradation in information timeliness underscores the critical influence that the data size has on the timeliness of the delivered content in the system. The comparative analysis of the five algorithms highlights that the proposed scheme consistently achieves a lower AoI across the tested range of data volumes. This finding suggests that the optimizations employed in the proposed method are effective at mitigating the AoI penalty incurred by the increased data size, offering superior performance in maintaining the timeliness of the transmitted information.

Accordingly, the proposed method performs better than the *PathGreedy*, *WoDDPG*, *WoCluster*, and *KRandom* schemes by roughly 8.2%, 73%, 87.5%, and 132%, respectively. Based on the numerical results, the multistage approach is effective. The result indicates that clustering IoTDs and applying the *DDPG-DHPSA* improves the system performance 87.5% and 73%, respectively.

Fig. 9 illustrates the effect of varying the data-generation intervals on the AoI performance for the proposed scheme and the four comparative approaches. The data-generation timestamps

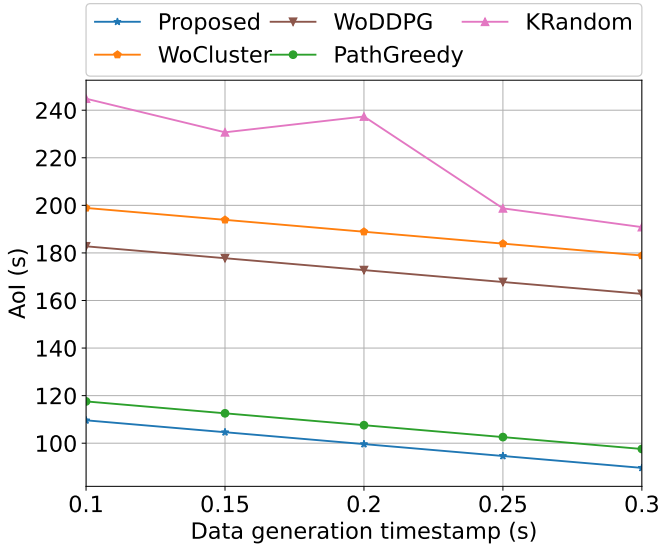
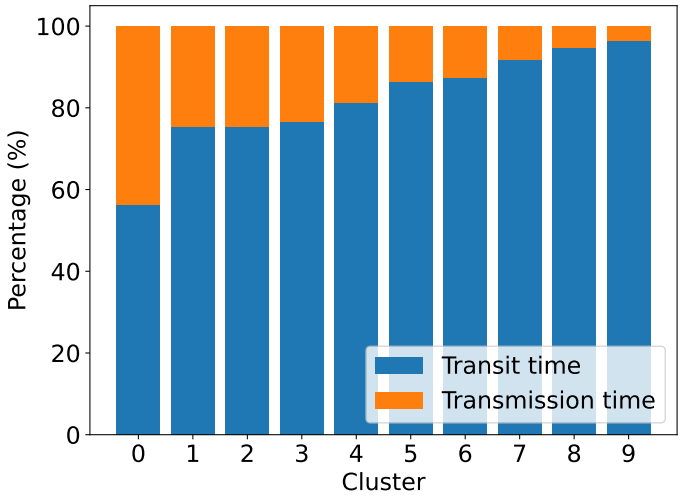


Figure 9: Performance by data-generation timestamp.

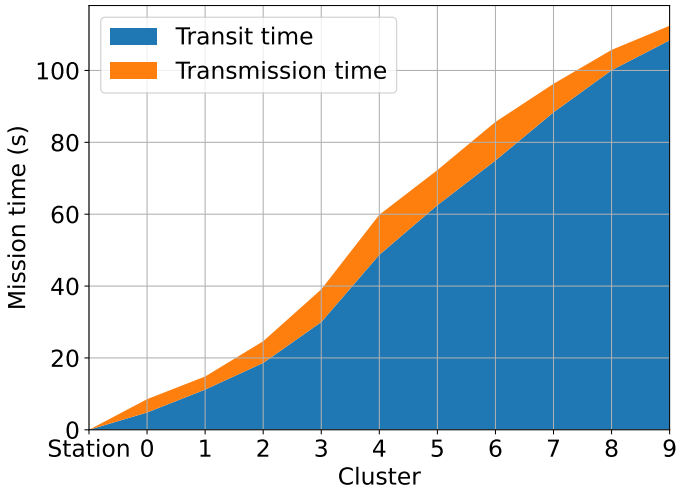
for the IoTDs were varied from 0.1 to 0.3 s to investigate the effect of delayed data generation on the AoI of the system. The results reveal that, as the data-generation intervals increase, the AoI decreases. The inverse relationship observed in the graph is attributed to the fact that a lower data-generation frequency would reduce the time that the data at the IoTDs spend waiting in the queue to be sent to the UAV. Therefore, the data waiting times are reduced, leading to a lower AoI at the destination. The proposed scheme significantly improves AoI performance, reducing it by an average of 4.91% for each 0.5-s increment. This significant enhancement highlights the effectiveness of the proposed approach in minimizing the AoI and ensuring the timely delivery of information from the IoTDs to the UAV. These results highlight how crucial it is to consider data-generation patterns.

Fig. 10 presents a comprehensive analysis of the mission time at each cluster, comprising two critical components: the transit time of the UAV and transmission time from the IoTDs to the UAV. Quantitatively, the mission time is dominated by the transmission time from the IoTDs to the UAV, consistently accounting for a greater proportion of time than the UAV transit time. The transmission time constitutes about 60% to 80% of the total mission time across all clusters, as presented in Fig. 10(a). This observation highlights the significant influence of the data transmission overhead on the overall mission efficiency and emphasizes the importance of optimizing the communication protocols between IoTDs and the UAV. Moreover, the clusters are sequentially labeled according to their order in the UAV trajectory planning.

Fig. 10(b) displays the distribution of the mission time across clusters. As the UAV progresses along its path, the mission duration monotonically increases from one cluster to the next. The explanation for this phenomenon is that the subsequent clusters are dependent on the completion of tasks at the previous clusters. Clusters further along the UAV trajectory are subjected to longer mission times due to the compounding effect of waiting for the UAV to traverse and complete its tasks at all prior clusters. This sequential nature of the UAV operation introduces a cumulative



(a) Percentage of transit and transmission times



(b) Value of transit and transmission times

Figure 10: Mission time allocation.

delay that prolongs the mission duration for clusters encountered later in the trajectory. For instance, Cluster 1 must wait until the UAV transits from the station to Cluster 0, followed by the transit from Cluster 0 to 1, once the data transmission in Cluster 0 is complete. This approach creates a bias in the transmission and transit times that affects the mission time. The transmission time and trajectory of the UAV are attractive areas of research for ensuring fairness requirements in systems with varying data demands.

Fig. 11 investigates the AoI at each cluster by computing the statistics on the AoI in 100 states. The AoI for clusters increases due to their order in the trajectory, as clusters farther away in the trajectory require more time for mission completion. In addition, HPs and the UAV trajectory are updated as the data demands and IoTD distributions change, creating variations in the AoI of each cluster. This finding implies that the proposed approach is suitable for various scenarios.

6. Conclusion

This paper investigated the challenges of optimizing the UAV trajectory in data collection in IoT networks, focusing

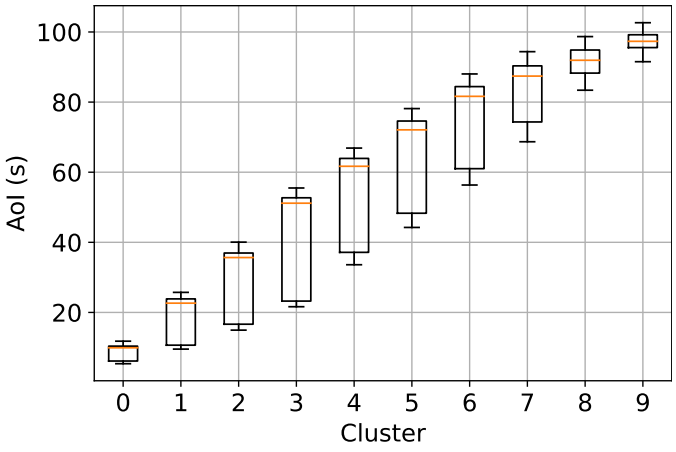


Figure 11: Age of information statistics by cluster.

on minimizing the AoI and mission time in an SCMA-based UAV-assisted network architecture. To address the complexity of the problem, we formulated it as a mixed-integer non-convex optimization problem, which was transformed into an MDP to capture the dynamic nature of the system. The proposed solution approach comprises two critical components. First, we employed a k-means dynamic clustering algorithm to group IoTDs based on geographical proximity. Then, we developed the DDPG-DHPSA to identify the optimal HPs in each cluster. The extensive simulation results demonstrated that the proposed framework significantly outperforms the baseline approaches: WoCluster, WoDDPG, PathGreedy, and KRandom. The implementation of k-means dynamic clustering and the DDPG-DHPSA achieved about 87.5% and 73% improvements in system performance, respectively. These results underscore the effectiveness of the proposed approach in optimizing the UAV trajectory, minimizing the AoI, and reducing the mission time. Future research directions include exploring the deployment of multiple collaborative UAVs to cover larger areas and addressing the fairness problems related to the mission time allocation over the clusters. In addition, a comprehensive analysis of the solution in different multiple access environments should be studied.

References

[1] E. Esenogho, K. Djouani, A. M. Kurien, Integrating artificial intelligence Internet of Things and 5G for next-generation smartgrid: A survey of trends challenges and prospect, IEEE Access 10 (2022) 4794–4831.

[2] E. El Haber, H. A. Alameddine, C. Assi, S. Sharafeddine, UAV-aided ultra-reliable low-latency computation offloading in future IoT networks, IEEE Transactions on Communications 69 (10) (2021) 6838–6851.

[3] D. S. Lakew, A.-T. Tran, A. Masood, N.-N. Dao, S. Cho, A Review on AI-Driven Aerial Access Networks: Challenges and Open Research Issues, in: 2023 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), IEEE, 2023, pp. 718–723.

[4] S.-M. Tseng, W.-Y. Chen, Cross-layer codebook allocation for uplink SCMA and PDNOMA-SCMA video transmission systems and a deep learning-based approach, IEEE Systems Journal 17 (1) (2022) 294–305.

[5] Q. Li, G. Liu, Y. Zhao, C. Liu, F. Xu, Deep Learning-Based Downlink OTFS-SCMA, IEEE Wireless Communications Letters 13 (2024) 3434–3438.

[6] P. Liu, K. An, J. Lei, Y. Sun, W. Liu, S. Chatzinotas, Computation rate maximization for SCMA-aided edge computing in IoT networks: A multi-

agent reinforcement learning approach, IEEE Transactions on Wireless Communications 23 (2024) 10414–10429.
[7] D. Li, S. Xu, C. Zhao, Y. Wang, R. Xu, B. Ai, Data Collection In Laser-Powered UAV-Assisted IoT Networks: Phased Scheme Design Based on Improved Clustering Algorithm, IEEE Transactions on Green Communications and Networking (2023).
[8] Y. Zhu, S. Wang, Efficient aerial data collection with cooperative trajectory planning for large-scale wireless sensor networks, IEEE Transactions on Communications 70 (1) (2021) 433–444.
[9] X. Song, X. Fu, M. Ren, P. Pace, G. Aloi, G. Fortino, Prediction-based data collection of UAV-assisted Maritime Internet of Things, Vehicular Communications (2024) 100854.
[10] X. Fu, X. Huang, Q. Pan, P. Pace, G. Aloi, G. Fortino, Cooperative data collection for UAV-assisted maritime IoT based on deep reinforcement learning, IEEE Transactions on Vehicular Technology 73 (7) (2024) 10333–10349.
[11] X. Fu, C. Deng, A. Guerrieri, Low-AoI data collection in integrated UAV-UGV-assisted IoT systems based on deep reinforcement learning, Computer Networks (2025) 111044.
[12] M. Samir, C. Assi, S. Sharafeddine, D. Ebrahimi, A. Ghrayeb, Age of information aware trajectory planning of UAVs in intelligent transportation systems: A deep learning approach, IEEE Transactions on Vehicular Technology 69 (11) (2020) 12382–12395.
[13] 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 Internet of Things Journal 9 (7) (2021) 5216–5228.
[14] M. Sun, X. Xu, X. Qin, P. Zhang, AoI-energy-aware UAV-assisted data collection for IoT networks: A deep reinforcement learning method, IEEE Internet of Things Journal 8 (24) (2021) 17275–17289.
[15] J. Zhang, K. Kang, M. Yang, H. Zhu, H. Qian, Aoi-minimization in uav-assisted IoT network with massive devices, in: 2022 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2022, pp. 1290–1295.
[16] G. Zhu, L. Guo, C. Dong, X. Mu, Mission time minimization for multi-UAV-enabled data collection with interference, in: 2021 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2021, pp. 1–6.
[17] Y. Xu, T. Zhang, J. Loo, D. Yang, L. Xiao, Completion time minimization for UAV-assisted mobile-edge computing systems, IEEE Transactions on Vehicular Technology 70 (11) (2021) 12253–12259.
[18] M. Li, S. He, H. Li, Minimizing mission completion time of UAVs by jointly optimizing the flight and data collection trajectory in UAV-enabled WSNs, IEEE Internet of Things Journal 9 (15) (2022) 13498–13510.
[19] X. Gao, X. Zhu, L. Zhai, Minimization of aerial cost and mission completion time in multi-UAV-enabled iot networks, IEEE Transactions on Communications (2023).
[20] T. Khurshid, W. Ahmed, M. Rehan, R. Ahmad, M. M. Alam, A. Radwan, A DRL Strategy for Optimal Resource Allocation Along With 3D Trajectory Dynamics in UAV-MEC Network, IEEE Access (2023).
[21] A. Chapnevis, E. Bulut, AoI-optimal cellular-connected UAV trajectory planning for IoT data collection, in: 2023 IEEE 48th Conference on Local Computer Networks (LCN), IEEE, 2023, pp. 1–6.
[22] A. Chapnevis, E. Bulut, UAV Mesh Network Trajectory Planning for Age Optimal Data Collection in Infrastructureless Areas, in: ICC 2024-IEEE International Conference on Communications, IEEE, 2024, pp. 1563–1568.
[23] A. Chapnevis, E. Bulut, Time-efficient approximate trajectory planning for AoI-centered multi-UAV IoT networks, Internet of Things 29 (2025) 101461.
[24] K. Liu, J. Zheng, UAV trajectory optimization for time-constrained data collection in UAV-enabled environmental monitoring systems, IEEE Internet of Things Journal 9 (23) (2022) 24300–24314.
[25] X. Liu, H. Liu, K. Zheng, J. Liu, T. Taleb, N. Shiratori, AoI-minimal clustering, transmission and trajectory co-design for UAV-assisted WPCNs, IEEE Transactions on Vehicular Technology 74 (2025) 1035–1051.
[26] H. Yin, N. Li, J. Guo, J. Zhu, X. She, NR Coverage Enhancements for PUSCH, IEEE Communications Magazine 60 (7) (2022) 36–42.
[27] K. P. Sinaga, M.-S. Yang, Unsupervised K-means clustering algorithm, IEEE access 8 (2020) 80716–80727.
[28] T. P. Truong, H. V. Nguyen, N.-N. Dao, W. Noh, S. Cho, Orthogonalized rsma-based flexible multiple access in digital twin edge networks, IEEE Transactions on Wireless Communications 23 (12) (2024) 18740–18756. doi:10.1109/TWC.2024.3476383.
[29] O. A. Amodu, C. Jarray, R. A. R. Mahmood, H. Althumali, U. A. Bukar, R. Nordin, N. F. Abdullah, N. C. Luong, Deep reinforcement learning for aoi minimization in uav-aided data collection for wsn and iot: A survey, IEEE Access (2024).
[30] T. P. Truong, T. M. T. Nguyen, T.-V. Nguyen, N.-N. Dao, S. Cho, RSMA for Uplink MIMO Systems: DRL-Based Achievable System Sum Rate Maximization, in: 2023 IEEE Globecom Workshops (GC Wkshps), IEEE, 2023, pp. 878–883.
[31] T. P. Truong, V. D. Tuong, N.-N. Dao, S. Cho, Flyreflect: Joint flying irs trajectory and phase shift design using deep reinforcement learning, IEEE Internet of Things Journal 10 (5) (2023) 4605–4620. doi:10.1109/JIOT.2022.3218740.
[32] K. Ikeagu, Y. Ding, C. Song, M. R. Khandaker, Intelligent Reflecting Surface Optimization for MIMO Communication Using Deep Reinforcement Learning, in: 2023 31st Telecommunications Forum (TELFOR), IEEE, 2023, pp. 1–4.
[33] S. Lloyd, Least squares quantization in pcm, IEEE Transactions on Information Theory 28 (2) (1982) 129–137. doi:10.1109/TIT.1982.1056489.
[34] T. Phung Truong, T. My Tuyen Nguyen, T. Vi Nguyen, N.-N. Dao, S. Cho, Energy efficiency in rsma-enhanced active ris-aided quantized downlink systems, IEEE Journal on Selected Areas in Communications 43 (3) (2025) 834–850. doi:10.1109/JSAC.2025.3531522.
[35] X. Gao, X. Zhu, L. Zhai, Aoi-sensitive data collection in multi-uav-assisted wireless sensor networks, IEEE Transactions on Wireless Communications 22 (8) (2023) 5185–5197.