Joint Resource Allocation and Trajectory Planning in Air-Ground Collaborative Edge Computing Power Offloading Network

Meng Gu, *Student Member, IEEE,* Yaxi Liu, *Member, IEEE,* Xulong Li, *Student Member, IEEE,* Jiahao Huo, *Member, IEEE,* Wei Huangfu, *Member, IEEE,*

***Abstract*—Unmanned aerial vehicle (UAV)-enabled computation offloading network where the internet of things (IoT) devices transmit to UAVs to help further process data. However, there are still some limitations in the existing UAV-enabled computation offloading network. They merely consider homogeneous aerial nodes to help offload computing tasks. This scheme faces the risk of air link damage, such as visibility conditions, and energy depletion. Motivated by this, we propose an edge computing power offloading network where not only the aerial nodes but also the ground nodes cooperatively help computation offloading from IoT devices and mobile devices. We establish an optimization with the goal of the task offloading failure penalty to optimize the UAV/vehicle trajectories, task offloading ratio, task association indicator, transmit power of IoT devices, and the CPU frequency of IoT devices and mobile devices, subject to boundary constraints, energy constraints, and power constraints. Such an optimization is complicated with complex objective functions, and various optimization variables are coupled, which increases the difficulty of solving this problem. We convert this optimization into a Markov decision process (MDP), and propose two state-of-the-art deep reinforcement learning (DRL) algorithms, say the soft actor-critic (SAC) and proximal policy optimization (PPO), to accurately and efficiently solve this MDP. Experiments show the convergence and effectiveness of the proposed framework in edge computing power offloading networks. In addition, the proposed SAC and PPO outperform the other state-of-the-art algorithms. Furthermore, the newly introduced auxiliary vehicles significantly reduce the task offloading failure penalty.**

***Index Terms*—Unmanned aerial vehicle (UAV), internet of things (IoT), trajectory planning, computation offloading.**

I. Introduction

With the deep integration of 6th generation mobile communication technology (6G) networks and extended reality (XR), and the exponential growth of internet of things (IoT) devices, computing-intensive applications represented by immersive interaction and real-time autonomous decision-making are driving a paradigm shift in communication systems in terms of latency , reliability, and energy efficiency. However, devices are limited by battery capacity and computational power, making it difficult for them to handle such tasks independently, giving rise to computation offloading (CO) technology [1] [2]. This technology significantly reduces terminal energy consumption and improves quality of service (QoS) by migrating tasks to edge servers or the cloud. Nevertheless, the traditional ground-based station (GBS)-based offloading scheme faces two major challenges [3]. First, GBS deployment is fixed, making it difficult to dynamically adapt to unexpected traffic or demand in remote areas; second, in dense urban areas or complex terrain, signals are easily blocked by buildings, resulting in unstable communication links [4].

Unmanned aerial vehicle (UAV) assisted computation offloading provides an innovative way to solve the above problems: with its high mobility, on-demand deployment capability, and line-of-sight (LoS) channel advantages, UAVs can quickly fly to mission hotspots to build temporary edge computing nodes, and also optimize the quality of the user-UAV channel by flexibly adjusting the flight trajectory and altitude [5] [6] [7]. It can also optimize user-UAV channel quality by flexibly adjusting flight trajectory and altitude. In disaster relief or temporary large-scale activities, UAVs can replace damaged or missing ground infrastructure. Earlier studies focused on the convergence of terrestrial base stations and mobile edge computing (MEC). For example, Mao et al [8] proposed a delay minimization offloading framework based on GBS, which reduces the end-to-end delay by jointly optimizing the task allocation and computational resources. However, such approaches are limited by the static deployment of GBS and are difficult to adapt to dynamic environment requirements. In recent years, UAV-assisted computation offloading has become a research hotspot due to its unique advantages. Dang et al [9] constructed a trade-off model between UAV energy consumption and task latency and proposed a dynamic trajectory planning algorithm based on reinforcement learning. Gu et al [10] proposed a lightweight encryption offloading protocol based on physical layer security to address the vulnerability of UAV-user channels to eavesdropping.

However, there are still some limitations in the existing UAV-enabled computation offloading. They primarily consider homogeneous aerial nodes to assist in offloading computing tasks. This approach faces several significant challenges, such as the potential for air link disruption due to environmental

Corresponding author: Wei Huangfu.

This work was supported by the National Natural Science Foundation of China (Grant No. U22A2005, 62301028).

Meng Gu, Yaxi Liu, Xulong Li, Jiahao Huo, and Wei Huangfu are with the School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China.(e-mail: lemongu778@gmail.com; yaxi.ustb@gmail.com; 2219387653@qq.com; huojiahao@ustb.edu.cn; huangfuwei@ustb.edu.cn).

ISBN 978-3-903176-72-0 © 2025 IFIP

factors like visibility conditions, unpredictable weather, and interference. Additionally, the risk of energy depletion in aerial nodes is a concern, as the limited battery life can impact the reliability and efficiency of the system. These limitations make the scheme vulnerable to potential failures in dynamic or demanding scenarios, where uninterrupted service and high performance are crucial.

Motivated by this, we study an edge arithmetic network in which a set of UAVs and a set of vehicles act as air-ground edge servers for IoT devices. We establish a multi-objective optimization model with the goal of the task offloading failure penalty to optimize the UAV/vehicle trajectories, task offloading ratio, task association indicator, transmit power of IoT devices or mobile devices, and the CPU frequency of IoT devices or mobile devices, subject to boundary constraint, energy constraints, and power constraints. To solve the multi-dimensional joint optimization challenge, this study innovatively models the problem as a multi-intelligence markov decision process (MDP) under incomplete information. Based on the deep reinforcement learning framework, an improved flexible actor-critic (SAC) algorithm and proximal policy optimization (PPO) are proposed to enable UAVs or unmanned vehicles, and IoT devices to autonomously decide the optimal computational offloading strategies and movement trajectories. Experimental results demonstrate the convergence and efficiency of the proposed framework in edge computing power offloading networks. Moreover, the performance of the proposed SAC and PPO algorithms surpasses that of other state-of-the-art methods. Additionally, the introduction of auxiliary vehicles plays a crucial role in significantly reducing the penalty associated with task-offloading failures.

Next, we will introduce the system model construction, algorithm design and optimization, and simulation experiments and performance analysis respectively.

II. System Model

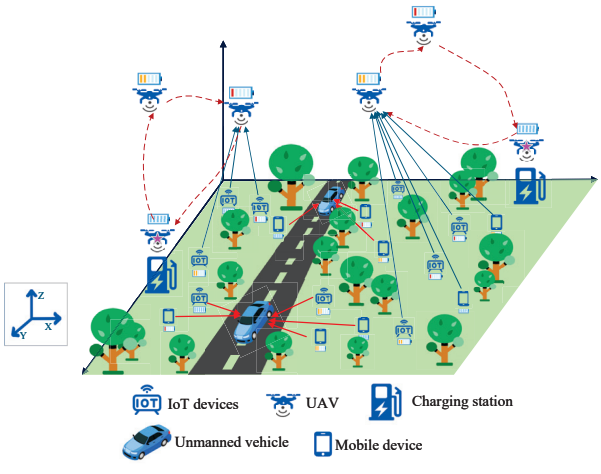

Fig. 1. The multi-UAV cooperative vehicle edge arithmetic network architecture system.

We consider a multi-UAV cooperative vehicle edge arithmetic network architecture system. The architecture is shown in Fig.1. This architecture contains three types of core units: clusters of IoT devices and mobile devices, clusters of UAVs with onboard edge servers, and mobile edge computing vehicles. The system device layer consists of I IoT devices and M mobile terminals, the control layer deploys $\mathbf{K} = \{1, ..., K\}$ UAVs equipped with edge servers, and the mobile infrastructure layer consists of $\mathbf{L} = \{1, ..., L\}$ intelligent vehicles with computational capabilities consisting of. This architecture is located in a region R. The network coverage is a square area R with side length L_{max}, which traditional terrestrial base stations do not serve. We consider that the terminal devices offload some of the computational tasks to UAVs or vehicles for processing, and each task can be offloaded to at most one UAV or one vehicle.

Considering that IoT devices and mobile devices have homogeneity in the dimension of computational offloading, in order to simplify the model complexity, we adopt IoT devices as a typical research object for modeling and optimization solutions uniformly. We use a discrete-time model that divides the decision time into N equal time slots, the set of which is defined as $\mathbf{N} = \{1, ..., N\}$. In time slot n, the ith IoT device generates a computation task denoted as $CT_{i,n} = \left\{D_{i,n}, C_{i,n}^{CPU}, TH_{i,n}^{D}\right\}$, where $D_{i,n}$ is the data size of the computation task, $C_{i,n}^{CPU}$ is the number of CPU cycles required for the task to compute 1 bit, i.e., the computation density (in cycles/bit), $TH_{i,n}^{D}$ is the maximum tolerable delay (in s), i is any value in $\mathbf{I} = \{1, ..., I\}$ and n is any value in $\mathbf{N} = \{1, ..., N\}$.

A. Communication Model

Considering the spatial dependence of air-ground and ground-ground communication links, we establish a three-dimensional Cartesian coordinate system in the target area. The position of the IoT device is denoted as $\boldsymbol{u}_{i,n}^{C} = \{x_{i,n}^{C}, y_{i,n}^{C}, z_{i,n}^{C}\}$. $x_{i,n}^{C}$ denotes the horizontal coordinate of the i^{th} IoT device at time slot n, $y_{i,n}^{C}$ denotes the vertical coordinate of the i^{th} IoT device at time slot n, and $z_{i,n}^{C}$ denotes the height of the i^{th} IoT device C_i at time slot n. The position of the k^{th} UAV U_k is denoted as $\boldsymbol{u}_{k,n}^{U} = \{x_{k,n}^{U}, y_{k,n}^{U}, z_{k,n}^{U}\}$. All UAVs can fly in the Z_{min} to Z_{max} altitude range over the target area. The position of the l^{th} vehicle V_l is denoted as $\boldsymbol{u}_{k,n}^{V} = \{x_{1,k,n}^{V}, y_{1,k,n}^{V}, z_{1,k,n}^{V}\}$. The horizontal distance between the i^{th} IoT device C_i and the k^{th} UAV U_k can be expressed as:

$$d_{i,k,n} = \sqrt{\left(x_{k,n}^{U} - x_{i,n}^{C}\right)^2 + \left(y_{k,n}^{U} - y_{i,n}^{C}\right)^2}. \quad (1)$$

The horizontal distance between the i^{th} IoT device C_i and the l^{th} vehicle V_l can be expressed as:

$$d_{i,l,n} = \sqrt{\left(x_{k,n}^{V} - x_{i,n}^{C}\right)^2 + \left(y_{k,n}^{V} - y_{i,n}^{C}\right)^2}. \quad (2)$$

For the i^{th} IoT device C_i to be able to offload computational tasks to the UAV or vehicle, the distance should meet certain conditions.

$$d_{i,j,n} \leq D_{max}, \quad (3)$$

where D_{max} is maximum communication distance.

In this system, the vehicle and the IoT device are usually in a low obstacle density scenario, so we consider the line-of-sight (LoS) link between the IoT device and the UAV, vehicle. The channel power gain between the i^{th} IoT device C_i and the k^{th} UAV U_k or l^{th} vehicle V_l can be expressed as

$$h_{i,j,n} = \frac{G_j}{d_{i,j,n}^{\kappa_j} + Z_{j,n}^2}, \tag{4}$$

where j can take values from **L** and **K**. When j is taken from **L**, κ_l is 2.1 and G_l is 5 dBi. $Z_{l,t}$ satisfies the Rayleigh decay distribution [11]. When j is taken from **K**, κ_k is 2.2 and G_k is 3 dBi. $Z_{k,t}$ satisfies the lognormal decay distribution.

For ease of description, we denote $\alpha_{i,j,n} \in \{0,1\}$ as whether the device offloads the task to U_k or V_l in time slot n. $\alpha_{i,j,n} = 1$ means that the IoT terminal device chooses to perform computational offloading in time slot n, otherwise $\alpha_{i,j,n} = 0$. The data rate $R_{i,j,n}$ between the i^{th} IoT device C_i and the k^{th} UAV U_k or l^{th} vehicle V_l can be expressed as

$$R_{i,j,n}^C = \frac{B_s}{\sum_{i'=1}^{I} \alpha_{i',j,n}} \log_2\left(1 + \frac{P_{i,j,n} h_{i,j,n}}{I_{i,j,n} + \sigma^2}\right), \tag{5}$$

where σ^2 is the Gaussian noise power, B_s is the available bandwidth, $\alpha_{i',j,n} \in \{\alpha_{i,j,n}^U, \alpha_{i,j,n}^V\}$, $P_{i,j,n}$ is the transmission power of C_i, $\sum_{i'=1}^{I} \alpha_{i',j,n}$ is the number of IoT devices associated with U_k or V_l, and $I_{i,j,n}$ is interference from IoT devices that select other UAVs or vehicles, which can be written as

$$I_{i,j,n} = \sum_{a=1,a\neq k}^{K} \sum_{b=1,b\neq i}^{I} \alpha_{a,b,n} P_{a,b,n} h_{a,b,n}. \tag{6}$$

B. System Metrics

The IoT device offloads part of the computation task to the UAV or vehicle and computes the rest locally. We set the local computation task and the offloading task to be generated and executed in parallel at the same time, then the total delay can be expressed as

$$T_{i,n} = \max(T_{i,n}^{Local}, T_{i,n}^{Trans} + T_{i,n}^{UAV}). \tag{7}$$

The delay caused by processing the computational task should be no greater than its maximum delay $T_{i,max}$.

The transmission delay depends on the amount of data being offloaded and the data rate being transmitted. The delay of the IoT device C_i to send the offloaded computational task to U_k or V_l is

$$T_{i,n}^{Trans} = \frac{\beta_{i,n} D_{i,n}}{R_{i,j,n}^C}. \tag{8}$$

The local computation delay $T_{i,n}^{local}$ is determined by the amount of local computation and the speed of the computation. The amount of local computation is related to the size of the data of the computation task that is not offloaded to the UAV or the vehicle, and the speed of computation is related to the CPU frequency $f_{i,n}^{mob}$ of C_i and the number of CPU turns $C_{i,n}^{CPU}$ required to compute 1 bit for that task. The expression for the local computation latency is denoted as

$$T_{i,n}^{local} = \frac{(1-\beta_{i,n}) D_{i,n} C_{i,n}^{CPU}}{f_{i,n}^{mob}}, \tag{9}$$

where $\beta_{i,n}$ denotes the offload ratio of the computing task, $f_{i,n}^{mob} \in [0, f_{i,max}^{mob}]$, and $f_{i,max}^{mob}$ denotes the maximum CPU frequency of C_i.

The total energy consumption of the IoT device includes the energy consumption resulting from locally processed computational tasks and the energy consumption resulting from off-loaded computational tasks sent to an edge server on the UAV or in the vehicle. We can express the energy consumption as

$$EU_{i,n} = EU_{i,n}^{local} + EU_{i,n}^{Trans}. \tag{10}$$

The energy consumption of IoT device C_i is expressed as

$$E_{i,n}^{Trans} = P_{i,n}^{mob} T_{i,n}^{Trans}, \tag{11}$$

where $P_{i,n}^{mob}$ denotes the transmitted power of IoT device C_i.

The i^{th} IoT device C_i should not consume more energy than the initial energy. Otherwise, the computational task will not be processed or discarded. In time slot n+1 the initial energy is

$$EU_{i,n+1,0} = min\{EU_{i,n,0} - EU_{i,n}, 0\}. \tag{12}$$

Consumption of energy needs to have a condition such as

$$EU_{i,n} \leq EU_{i,n,0}. \tag{13}$$

To ensure the successful execution of the task, the processing success rate of the computational task has become a key metric of our concern. The processing success rate of the computational task of the i^{th} IoT device C_i from time slot 1 to time slot n can be expressed as

$$TH_{i,n}^C = \max\left\{ \begin{array}{l} EU_{i,n}^{TH} - EU_{i,n}, \\ \left(\sum_{k=1}^{K} \alpha_{i,k,n}^U + \sum_{l=1}^{L} \alpha_{i,l,n}^V\right) \mathbb{1}(ET_{i,n}^{TH} - T_{i,n}) \end{array} \right\}, \tag{14}$$

where $\mathbb{1}(A)$ is an indicator function that takes the value of 1 when A is true and 0 otherwise. $ET_{i,n}^{TH}$ denotes the corresponding threshold of the maximum delay and $EU_{i,n}^{TH}$ denotes the corresponding threshold of the maximum energy.

The communication penalty index ensures both communication performance and fairness among communication users.The data expression for the penalty function can be expressed as

$$p_{i,n}^C = \begin{cases} T_{i,n}/ET_{i,n}^{TH}, & TH_{i,n}^C = 1, \\ \min\left(p_{i,n-1}^C + 1, F^{\max}\right), & \text{Otherwise}, \end{cases} \tag{15}$$

where $F^C = \sum_{n=1}^{N} p_{i,n}^C$is the sum of the communication penalty index for all communication users. A smaller penalty index means more fairness among communication users and a wider communication range.

$$R_{i,n}^C = \begin{cases} R_{i,n}^{C,Max}, & \mathbb{1}(A_{i,n}) = 1, \\ \hat{R}_{j_{i,n},i,n}^{C,U\to C}, & (1-\mathbb{1}(A_{i,n}))\mathbb{1}(B_{i,n}) = 1, \\ 0, & \text{Otherwise}, \end{cases} \tag{16}$$

where $j_{i,n}$ is an index of the UAV or vehicle that performed the service. $\hat{R}_{k,i,n}^{C,U\to C}$ denotes the data rate from UAV, expressed as

$$R_{j_{i,n},i,n}^C = B_s \log_2\left(1 + p_{j_{i,n},n}^U h_{j_{i,n},i,n}/(I_{i,n}+\sigma^2)\right), \tag{17}$$

where $I_{i,n}$ denotes the interference term. σ^2 denotes the Gaussian white noise term. p_k^U denotes the transmit power of U_k.

C. Problem Formulation

We propose a multi-objective framework aimed at achieving three-dimensional performance enhancement of edge arithmetic networks. We represent the optimization problem as

$$\mathbf{P1}: \min_{\mathbb{A},\mathbb{B}} F^C \tag{18}$$

$$\text{s.t.} \quad \sum_{k=1}^{K} \alpha_{i,k,n}^U + \sum_{l=1}^{L} \alpha_{i,l,n}^V \le 1, \forall i \in \mathcal{I}, n \in \mathcal{N}, \tag{18a}$$

$$\sum_{i=1}^{I}\left(\sum_{k=1}^{K} \alpha_{i,k,n}^U + \sum_{l=1}^{L} \alpha_{i,l,n}^V\right) \le I, \forall n \in \mathcal{N}, \tag{18b}$$

$$P_{i,k,t}^U \le P_i^{max}, \forall i \in \mathcal{I}, k \in \mathcal{K}, \tag{18c}$$

$$P_{i,l,t}^V \le P_i^{max}, \forall i \in \mathcal{I}, l \in \mathcal{L}, \tag{18d}$$

$$EU_{i,n} \le EU_{i,n,0}, \forall i \in \mathcal{I}, n \in \mathcal{N}, \tag{18e}$$

$$f_{i,n}^{\text{mob}} \le f_{i,\max}^{\text{mob}}, \forall i \in \mathcal{I}, n \in \mathcal{N}, \tag{18f}$$

$$\sum_{i=1}^{I} \alpha_{i,k,n}^U f_{i,n}^{\text{uav}} \le f_{\text{uav}}, \forall k \in \mathcal{K}, n \in \mathcal{N}, \tag{18g}$$

$$\sum_{i=1}^{I} \alpha_{i,l,n}^V f_{i,n}^{\text{veh}} \le f_{\text{veh}}, \forall l \in \mathcal{L}, n \in \mathcal{N}, \tag{18h}$$

$$-x_{\max} \le x_{k,n}^U \le x_{\max}, \forall k \in \mathcal{K}, n \in \mathcal{N}, \tag{18i}$$

$$-y_{\max} \le y_{k,n}^U \le y_{\max}, \forall k \in \mathcal{K}, n \in \mathcal{N}, \tag{18j}$$

$$Z_{\min} \le z_{k,n}^U \le Z_{\max}, \forall k \in \mathcal{K}, n \in \mathcal{N}, \tag{18k}$$

$$-x_{\max} \le x_{l,n}^V \le x_{\max}, \forall l \in \mathcal{L}, n \in \mathcal{N}, \tag{18l}$$

$$-y_{\max} \le y_{l,n}^V \le y_{\max}, \forall l \in \mathcal{L}, n \in \mathcal{N}, \tag{18m}$$

$$Z_{\min} \le z_{l,n}^V \le Z_{\max}, \forall l \in \mathcal{L}, n \in \mathcal{N}, \tag{18n}$$

where A is expressed as

$$\mathbb{A} = \left\{ \begin{matrix} \beta_{i,n},\ f_{i,n},\ P_{i,n},\ \alpha_{i,1,n}^U, \ldots, \alpha_{i,K,n}^U, \\ \alpha_{i,1,n}^V,\ \ldots, \alpha_{i,L,n}^V. \end{matrix} \right\}. \tag{18}$$

$\mathbb{A}$ is a computing task offloading decision variable for the i^{th} IoT device C_i. $\mathbb{B} = \{\mathbb{Q}_1, \mathbb{Q}_2\}$ is a decision variable for trajectory planning for UAVs and vehicles, where $\mathbb{Q}_1 = \{x_{k,n}^U, y_{k,n}^U, z_{k,n}^U\}$ and $\mathbb{Q}_2 = \{x_{l,n}^V, y_{l,n}^V, z_{l,n}^V\}$.

III. Problem Reconstruction and Proposed Solution

Such an optimization is complicated with complex objective functions, and various optimization variables are coupled, which increases the difficulty of solving this problem. We convert this optimization into a MDP, and propose two state-of-the-art DRL algorithms, say the SAC and PPO, to accurately and efficiently solve this MDP.

A. Problem Reformulation

P1 is essentially a MINLP problem, traditional optimization methods often fail due to high computational complexity under real-time requirements. To this end, we formally characterize P1 by MDP and place it in a multi-intelligent reinforcement learning (MARL) framework to achieve dynamic policy optimization and employ the SAC algorithm in Deep Reinforcement Learning to solve the problem. Thus, the adaptability defect and decision delay problem of traditional methods in time-varying environments are circumvented.

We reconstruct the problem as an MDP, represented as a four-element tuple $(\mathbb{S}, \mathbb{A}, \mathbb{P}, \mathbb{R})$, and we introduce the basic elements of the MDP.

Agent: we set all UAVs and unmanned vehicles in the area as intelligence that determine all variables.

Environment: the environment is a scenario that considers the multi-UAV cooperative vehicular edge arithmetic network.

State ($\mathbb{S}$): the set of state space within time slot n includes the position information of all vehicles $\{x_{l,n}^V, y_{l,n}^V, z_{l,n}^V\}, \forall l$, the position information of all UAVs $\{x_{k,n}^U, y_{k,n}^U, z_{k,n}^U\}, \forall k$, the position information of all IoT devices $\{x_{i,n}^C, y_{i,n}^C, z_{i,n}^C\}, \forall i$, and the energy consumption of all IoT devices $\{EU_{i,n}, \forall i\}$.

Action ($\mathbb{A}$): the set of actions within time slot n contains all the UAVs' flight distances$\{|x_{k,n}^U - x_{k,n-1}^U|, |y_{k,n}^U - y_{k,n-1}^U|, |z_{k,n}^U - z_{k,n-1}^U|, \forall k\}$, the vehicles' movement distances$\{|x_{l,n}^V - x_{l,n-1}^V|, |y_{l,n}^V - y_{l,n-1}^V|, |z_{l,n}^V - z_{l,n-1}^V|, \forall l\}$, transmit power$\{p_{i,k,n}^U, p_{i,l,n}^V | \forall i, k, l\}$, the set of associated indicators$\{\alpha_{i,k,n}^U, \alpha_{i,l,n}^V | \forall i, k, l\}$,the offload ratio of the computing task $\{\beta_{i,n} | \forall i\}$ and frequency of the all IoT devices$\{f_{i,n}^{mob} | \forall i\}$.

Policy ($\mathbb{P}$): The $\mathbb{P} = \mathbb{P}(\mathbb{S}_{n+1}|\mathbb{S}_n, \mathbb{A})$ represents the probability of moving to a new state $\mathbb{S}_{n+1}$ after taking an action in the state$\mathbb{S}_n$.

Reward ($\mathbb{R}$): the reward function evaluates the performance of the action taken by the agent in a given observation. The reward $\mathbb{R}_n$ is $-\tau F + a$ in the time slot n, where τ is a proportionality constant and a is a bias constant. When the constraints in the optimization problem cannot be satisfied, $\mathbb{R}_n$ is $\mathbb{R}_n - b$, where b is the penalty term. We set b to a sufficiently large value.

The goal is usually to maximize cumulative rewards, so solving the above MDP can eventually solve P1.

B. SAC Algorithm

We use the SAC algorithm to solve the MDP. The SAC algorithm achieves a balance between efficient exploration and robust strategy convergence in complex dynamic scenes by integrating the synergistic mechanism of maximum entropy theoretical framework and actor-critic architecture for the optimal control problem in continuous action space. There are five neural networks in the SAC algorithm, including a policy network $\pi(\mathbb{A}|\mathbb{S})$, two Q-value networks Q_{θ_1} and Q_{θ_2}, and two target Q-value networks $Q_{\theta_1^t}$ and $Q_{\theta_2^t}$.

We initialize the parameters of the neural network. Further, we interact with the environment to collect a number of state transfers $(\mathbb{S}_n, \mathbb{A}_n, \mathbb{R}_n, \mathbb{S}_{n+1})$ and store them in the experience playback pool.

1) **Update the Q-value network.** We use a randomly sampled batch of state transitions to update the network parameters. We update the two Q-value networks by minimizing loss functions. The loss functions of the two Q-value networks are

$$L_Q(\theta_1) = E\left[\left(y - Q_{\theta_1}(\mathbb{S}, \mathbb{A})\right)^2\right], \tag{19}$$

$$L_Q(\theta_2) = E\left[\left(y - Q_{\theta_2}(\mathbb{S}, \mathbb{A})\right)^2\right]. \tag{20}$$

y is the target value and the computational expression is

$$y = R + \gamma\left(\min_{i=1,2} Q_{\theta_i^t}(\mathbb{S}', \widetilde{\mathbb{A}}') - \varepsilon log\pi_\varphi(\widetilde{\mathbb{A}}'|\mathbb{S}')\right), \tag{21}$$

where $\mathbb{S}'$ denotes the new state, γ denotes the discount factor, φ denotes the temperature parameter, and the action $\mathbb{A}'$ is $\pi(\cdot|\mathbb{S}')$.
The Q-value network parameters are updated according to the following rules

$$\theta_1 = \theta_1 - \eta\nabla_{\theta_1} L_Q(\theta_1), \tag{22}$$

$$\theta_2 = \theta_2 - \eta\nabla_{\theta_2} L_Q(\theta_2), \tag{23}$$

where η denotes the learning rate.

2) **Update policy network**. The parameters of the policy network are updated by maximizing the expected return of the policy and the weighted sum of the entropy. The loss function is

$$L_\pi(\varphi) \tag{18}$$

$$= E\left[\left(\min_{i=1,2} Q_{\theta_i}(\mathbb{S}, \widetilde{\mathbb{A}}) - \varepsilon log\pi_\varphi(\widetilde{\mathbb{A}}|\mathbb{S})\right)^2\right]. \tag{24}$$

Policy network parameters can be updated in the following ways.

$$\varphi = \varphi + \eta\nabla_\varphi L_\pi(\varphi). \tag{25}$$

3) **Soft update target Q-value network**. The expressions for updating the two networks are

$$Q_{\theta_1^t} \leftarrow \eta Q_{\theta_1} + (1-\eta) Q_{\theta_1^t}. \tag{26}$$

$$Q_{\theta_2^t} \leftarrow \eta Q_{\theta_2} + (1-\eta) Q_{\theta_2^t}. \tag{27}$$

4) **Update** ε. Adaptive tuning by minimizing losses.

$$L_\varepsilon = E\left[-\varepsilon log\pi_\varphi(A|S) - \varepsilon\mathcal{H}\right], \tag{28}$$

where $\mathcal{H}$ denotes the target strategy entropy.

5) **Repeat steps**. We keep repeating the above steps until the algorithm converges.

C. Proximal Policy Optimization

PPO is a prominent reinforcement learning algorithm known for maintaining stable training processes and preventing performance degradation caused by excessive policy fluctuations during updates. As part of policy gradient methods, PPO directly optimizes the policy parameters by maximizing an objective function to achieve higher cumulative rewards.

The fundamental principle of PPO involves incorporating a proximity constraint during policy updates, which ensures the updated policy remains close to the previous one. Specifically, PPO employs a clipping mechanism on the probability ratio, representing the likelihood ratio of executing the same action under the new policy compared to the old policy. This clipping technique effectively prevents excessive deviations and thus preserves the stability of the training.

The main steps of the PPO algorithm can be briefly summarized as follows:

1) **Initialization**: Initialize the actor policy network (π_ω with parameter ω) and the critic value network (Q_ψ with parameter ψ).
2) **Data Collection**: Collect interaction data by running the current policy.
3) **Advantage Estimation**: To evaluate the accuracy of the value estimation, the Temporal Difference (TD) error is calculated. At a given time step n, the TD error is defined as

$$\eta_n = r_n + \gamma Q_\psi(\mathbb{S}_{n+1}) - Q_\psi(\mathbb{S}_n), \tag{29}$$

where Q_ψ denotes the value function.
To further reduce the variance and balance bias in the policy gradient estimation, generalized advantage estimation (GAE) is adopted

$$\hat{\mathbb{A}}_n(\gamma, \xi) = \sum_{m=0}^{\infty} (\gamma\xi)^m \eta_{n+m}, \tag{30}$$

where $\xi \in [0, 1]$ controls the bias-variance trade-off. Specifically, when $\xi = 0$, the GAE method simplifies to the one-step TD error; when $\xi = 1$, it approximates a Monte Carlo estimate.

4) **Policy Optimization**: Optimize the actor policy by maximizing a carefully designed objective function,

which includes the clipped ratio of probabilities, the estimated advantage, and an entropy term to encourage sufficient exploration:

$$L^{\text{clip}}(\omega) = \mathbb{E}_n\Big[\min\Big(\rho_n(\omega)\hat{\mathbb{A}}_n, \text{clip}(\rho_n(\omega), 1-\epsilon, 1+\epsilon)\hat{\mathbb{A}}_n\Big)\Big], \quad (31)$$

where $\rho_n(\omega) = \frac{\pi_\omega(\mathbb{A}_n|\mathbb{S}_n)}{\pi_{\omega_{\text{old}}}(\mathbb{A}_n|\mathbb{S}_n)}$, and ϵ is the clipping threshold.

5) **Probability Ratio Clipping**: Clip the probability ratio to ensure it remains within a predefined range, thus limiting policy changes and maintaining training stability.
6) **Updating the Value Network**: Formally, the loss function is expressed as

$$L^{\text{VF}}(\psi) = \left(Q_\psi(s_n) - \sum_{m=0}^{\infty}\gamma^m \frac{\pi_\omega(\mathbb{A}_{n+m}|\mathbb{S}_{n+m})}{\pi_{\omega_{\text{old}}}(\mathbb{A}_{n+m}|\mathbb{S}_{n+m})}\right)^2. \quad (32)$$

7) **Repeating the Training Procedure**: Iteratively repeat steps 2 through 6, continuously refining the actor and critic networks.

IV. Experiments and Discussion

In this section, we conducted a large number of simulation experiments. Firstly, the parameters in the simulation experiments are introduced, and then the effectiveness of the algorithm is verified by simulation experiments.

A. Simulation Scenario Setup

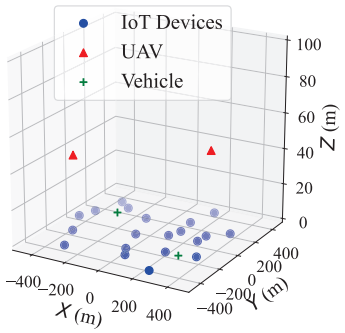

Fig. 2. The initial scene of the our system.

We set up a square target area with multiple IoT devices with mobile characteristics deployed along with it. The scenario is equipped with both UAVs and unmanned vehicles, as well as edge computing servers as base stations. Our initial scene is shown in Fig.2. The blue dots indicate IoT devices, the red triangles indicate drones, and the green plus signs indicate unmanned vehicles. Each IoT device generates an independent computation task during a time slot cycle. The tasks are executed with differential delay constraints, the successful tasks need to meet a random threshold of $20-40\,ms$, and the overtime tasks will be uniformly marked as $80\,ms$ delay records. The rest of the simulation parameters are detailed in Table I.

TABLE I
Core Parameter Configuration

Para.	Value	Para.	Value
Area length ($L_{\max}$)	1 km	Device count (I)	20
UAV count (K)	2	Vehicle count (L)	2
First UAV's position	[200, 300, 40]	Second UAV's position	[-250, -350, 50]
First vehicle's position	[280, -195, 0]	Second vehicle's position	[-290, 150, 0]
Task size	10–30 kb	IoT CPU freq. ($f_{i,\max}^{\text{mob}}$)	0.8 GHz
UAV CPU freq. (f_{uav})	2.0 GHz	Vehicle CPU freq. (f_{veh})	2.0 GHz
Delay constraint	20–40 ms	Timeout threshold	80 ms
Speed range	1–5 m/s	Time slot length	1 s
Min height ($Z_{\min}$)	100 m	Max height ($Z_{\max}$)	200 m
X boundary ($x_{\max}$)	500 m	Y boundary ($y_{\max}$)	500 m
Channel bandwidth	10 MHz	Noise power	10^{-9} W
Transmit power	1 W		

B. Experimental result

We obtained simulation results for different cases.

Fig.3 shows the 3D trajectories of the optimized UAVs and unmanned vehicles (auxiliary ground nodes) in the air-ground collaboration scenario. It can be observed in the figure that the UAV flexibly adjusts its flight path in the airspace according to the task distribution, realizing efficient air coverage and resource scheduling; at the same time, the UAV moves dynamically on the ground, assisting in completing the local computing task offloading and edge service supplementation. The trajectories of the two collaborative planning form a complementary relationship in space, reflecting the effect of air-ground collaborative optimization.

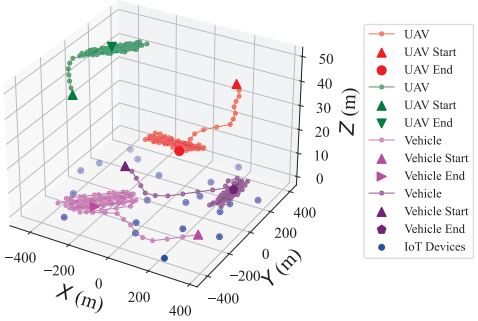

Fig. 3. The 3D trajectories of the optimized UAVs and unmanned vehicles.

Fig.4 shows the variation of Task offloading failure penalty with the number of iterations during the training process for different algorithms. It includes the PPO algorithm and SAC algorithm proposed and described in detail and also introduces the TD3 algorithm with the randomized strategy as the performance baseline.

From the simulation results, it can be observed that the performance of the randomized strategy is consistently poor. Our proposed PPO algorithm can quickly converge to a better performance level in the early stage of training, but its final convergence value is slightly inferior to that of the

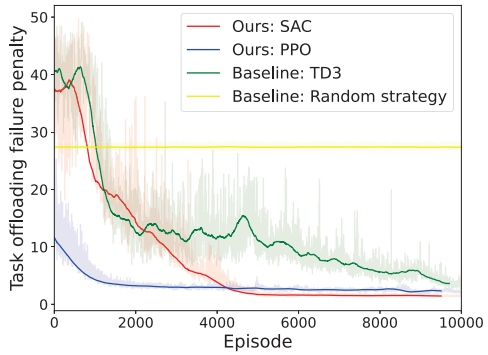

Fig. 4. Average cumulative penalty versus training episodes for different algorithms.

SAC algorithm; whereas the SAC algorithm, although initially converging slowly, gradually reaches and stabilizes at the optimal level of the task offloading failure penalty as the number of training times increases. In addition, we tested the performance of the TD3 algorithm in simulation.The TD3 algorithm converges poorly and ultimately fails to reach satisfactory optimization results. This simulation result further validates the effectiveness and reliability of our proposed PPO and SAC algorithms in this scenario.

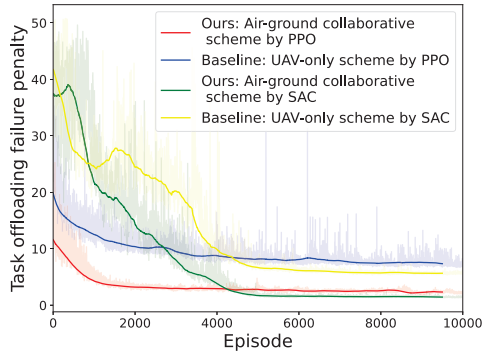

Fig. 5. Average cumulative penalty versus training episodes for different scenarios.

Fig.5 shows the simulation curves of the task offloading failure penalty with the number of training sessions for different scenarios. It contains two of our proposed air-ground collaboration scenarios (using PPO and SAC algorithms, respectively), as well as two baseline scenarios (using only UAVs, based on PPO and SAC algorithms, respectively). It is obvious from the figure that our proposed air-ground collaboration scenarios significantly outperform the UAV-only baseline scenarios in terms of task offload failure penalty metrics, showing the advantages of introducing ground-based auxiliary nodes.

In our proposed scenario, the scenario based on the PPO algorithm shows a faster initial convergence speed, but its final convergence value is not optimal. While the scenario based on the SAC algorithm shows a relatively slower initial convergence speed. However, the final task offloading failure penalty is minimized to achieve optimal convergence performance. Therefore, from an overall aspect, our proposed air-ground collaboration scenario can effectively reduce the risk of task offloading failure, and in particular, the SAC algorithm-based air-ground collaboration scenario exhibits the most optimal long-term performance.

So short-term emergency scheduling can prioritize PPO to achieve rapid deployment, while long-term adaptive optimization needs to combine the deep exploration capability of SAC, and the convergence speed and optimality can be taken into account in the future by using a hybrid training framework (initial PPO and late SAC).

V. Conclusion

Facing the contradiction between dynamic topology and sudden arithmetic demand in low-altitude economic scenarios, the air-ground cooperative edge arithmetic network architecture proposed in this paper realizes three-dimensional spatial elasticity arithmetic supply through the heterogeneous node synergy between UAVs and unmanned vehicles. Future work will focus on exploring the joint allocation model of three-dimensional spatial spectrum resources and computational resources and introducing a federated learning framework to strengthen the autonomous synergistic capability among nodes, in order to support the scaling of city-level low-altitude economic services.

References

[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] K. Sadatdiynov, L. Cui, L. Zhang, J. Z. Huang, S. Salloum, and M. S. Mahmud, "A review of optimization methods for computation offloading in edge computing networks," *Digit. Commun. Netw.*, vol. 9, no. 2, pp. 450–461, 2023.

[3] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong, "An air-ground integration approach for mobile edge computing in IoT," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 40–47, 2018.

[4] Y. Xu, T. Zhang, Y. Liu, D. Yang, L. Xiao, and M. Tao, "Uav-assisted mec networks with aerial and ground cooperation," *IEEE Trans. Wireless Commun.*, vol. 20, no. 12, pp. 7712–7727, 2021.

[5] Y. Du, K. Wang, K. Yang, and G. Zhang, "Energy-efficient resource allocation in UAV based MEC system for IoT devices," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM), 2018*. IEEE, 2018, pp. 1–6.

[6] Y. Liu, W. Huangfu, H. Zhou, H. Zhang, J. Liu, and K. Long, "Fair and energy-efficient coverage optimization for UAV placement problem in the cellular network," *IEEE Trans. Commun.*, vol. 70, no. 6, pp. 4222–4235, 2022.

[7] M. Gu, Y. Liu, B. He, W. Huangfu, and K. Long, "RIS-Assisted joint waveform design for PAPR-aware UAV-enabled integrated sensing and communication," *IEEE Trans. Veh. Technol.*, 2025.

[8] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[9] C. Deng, X. Fang, and X. Wang, "Uav-enabled mobile-edge computing for ai applications: Joint model decision, resource allocation, and trajectory optimization," *IEEE Internet Things J.*, vol. 10, no. 7, pp. 5662–5675, 2022.

[10] X. Gu, G. Zhang, M. Wang, W. Duan, M. Wen, and P.-H. Ho, "Uav-aided energy-efficient edge computing networks: Security offloading optimization," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4245–4258, 2021.

[11] "IEEE approved draft standard for wireless access in vehicular environments (WAVE) – multi-channel operation - corrigendum 1: Miscellaneous corrections," *IEEE Std 1609.4-2016/Cor1/D3, Apr. 2019*, pp. 1–12, 2019.