This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI10.1109/XXXXXXXXTBC, IEEE Communications Letters

Deep Reinforcement Learning Based Trajectory Design and Resource Allocation for UAV-Assisted Communications

Chiya Zhang, Zhukun Li, Chunlong He, Kezhi Wang and Cunhua Pan

***Abstract*—In this paper, we investigate the Unmanned Aerial Vehicles (UAVs)-assisted communications in three dimensional (3-D) environment, where one UAV is deployed to serve multiple user equipments (UEs). The locations and quality of service (QoS) requirement of the UEs are varying and the flying time of the UAV is unknown which depends on the battery of the UAVs. To address the issue, a proximal policy optimization 2 (PPO2)-based deep reinforcement learning (DRL) algorithm is proposed, which can control the UAV in an online manner. Specifically, it can allow the UAV to adjust its speed, direction and altitude so as to minimize the serving time of the UAV while satisfying the QoS requirement of the UEs. Simulation results are provided to demonstrate the effectiveness of the proposed framework.**

***Index Terms*—Unmanned aerial vehicles, Deep reinforcement learning, 3-D trajectory design, Uncertain flight time**

I. Introduction

Unmanned Aerial Vehicles (UAVs)-assisted communication is expected to play an important role in future wireless communications. There are many challenges that need to be addressed before UAVs can be effectively utilized for communication purposes [1]. The existing contributions on UAV-assisted communication systems can be divided into two categories: 1) UAV is statically deployed in the air to enhance wireless communication coverage; 2) UAVs are dynamically deployed, serving as the relay of the Base Station (BS) [2], and collecting Internet of Things (IoT) data, etc. Compared with the traditional ground BS, UAV-assisted-BS offeres the advantages such as enhanced mobility and increased likelihood of Line-of-Sight (LoS) communication with users. The existing contributions on dynamic deployment of UAV are normally based on: a) convex optimization algorithms [3]; and b) deep reinforcement learning (DRL) algorithms [4]. Compared with traditional convex optimization, DRL based algorithms may have higher performance and lower time complexity [5].

However, the above works assumed that the flying time/steps of the UAV is known and the UAV does not need to fly back to the starting point after its mission. Also, the locations of the user equipments (UEs) are static, which may not be applicable in the real-world systems.

Inspired by the works [6], [7], this paper presents the UAV-assisted communication system where the flying time of the UAV are unknown and the locations/requirement of the UE is varying. To address this problem, a proximal policy optimization 2 based (PPO2-based) low-complexity and on-line solution is proposed to minimize the flight time consumption of the UAV, while at the same time meeting other constants, i.e., completing communication tasks from the ground UEs.

This work was supported in part by the National Natural Science Foundation of China under Grant 62101161, in part by the Shenzhen Basic Research Program under Grant 20200812112423002, and 20200812112423002, in part by Horizon Europe COVER project under Grant 101086228.

(The Corresponding Authors are Zhukun Li, and Chunlong He)

C. Zhang and Z. Li are with the School of Electronic and Information Engineer-ing, Harbin Institute of Technology, Shenzhen 518055, China

C. He is with the Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518060, China

K. Wang is with Department of Computer Science, Brunel University London, Uxbridge, Middlesex, UB8 3PH

C. Pan is with the National Mobile Communication Research Laboratory, Southeast University, Nanjing 211189, China.

II. System Model

We consider a downlink communication model where one UAV serves N UEs with $\mathcal{N} = \{1, 2, ..., N\}$. The UAV has the same take-off and landing site, which can be represented as $\mathbf{q}_{ini} = (x_{ini}, y_{ini}, z_{ini})$. In this system, we assume that the UAV needs to complete all the tasks of the UEs and then return to the take-off site within the maximum flight time slot $T_{\max}$, otherwise it is seen as a failure. The goal of our model is to minimize the flight time slot t_{end} for the UAV to complete its mission and return to the take-off site. The coordinates of the UAV at the t-th time slot and the i-th UE can be represented as $\mathbf{q}[t] = (x_D[t], y_D[t], z_D[t]) \in \mathbb{R}^3$ and $\mathbf{u}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$, respectively. The limits of the UAV coordinates can be represented as

$$0 \leq x_D[t] \leq X_{\text{env}}, \forall t \in \mathcal{T}, \mathcal{T} = \{1, 2, ..., T_{\max}\}, \quad (1)$$

$$0 \leq y_D[t] \leq Y_{\text{env}}, \forall t \in \mathcal{T}, \quad (2)$$

$$z_D^{\min} \leq z_D[t] \leq Z_{\text{env}}, \forall t \in \mathcal{T}, \quad (3)$$

where $(X_{\text{env}}, Y_{\text{env}}, Z_{\text{env}})$ represents the size of the environment. $z_D^{\min}$ represents the minimum altitude for the UAV. At the begnining of a flight mission round, the location information of the UEs is generated randomly. In each time slot of a flight mission, the UEs move a distance in a random direction at a rate of 1m/s. It is assumed that $d_i[t]$ is the distance between the UAV and the i-th UE at the t-th time slot. Then we have

$$d_i[t] = \sqrt{(s_i[t]^2 + h_i[t]^2)}, \forall i \in \mathcal{N}, \forall t \in \mathcal{T}, \quad (4)$$

where $s_i[t]$ and $h_i[t]$ represent the horizontal distance and vertical distance between the UAV and the i-th UE in time slot t respectively. Then, the path loss can be given by

$$L(h_i[t], s_i[t]) = \frac{\eta_A}{1+\eta_a \exp(-\eta_b(\frac{180}{\pi}\arctan(\frac{h_i[t]}{s_i[t]}) - \eta_a))} + 20\log(d_i[t]^2) + \eta_B, \forall i \in \mathcal{N}, \forall t \in \mathcal{T}, \tag{5}$$

where $\eta_A = \eta_{LoS} - \eta_{NLoS}$, $\eta_B = \log(f_c) + 20\log(4\pi/c) + \eta_{NLoS}$, η_a and η_b are constants that depend on the environment [8], f_c is the carrier frequency (Hz), c is the speed of light (m/s), η_{LoS} and η_{NLoS} are the mean additional loss for LoS and NLoS links, respectively. $P^D[t]$ represents the transmission power of the UAV at the t-th time slot, and then the received power of the i-th UE $P_i^r[t]$ at the t-th time slot can be given by [8]

$$P_i^r[t] = P^D[t] - L(h_i[t], s_i[t]), \forall i \in \mathcal{N}, \forall t \in \mathcal{T}, \tag{6}$$

In order to guarantee the QoS between the UAV and the i-th UE, we assume that the received power of the i-th UE at the t-th time slot $P_i^r[t]$ should exceed a certain threshold $P_{\min}$ [9]

$$P_i^r[t] \geq P_{\min}, \forall i \in \mathcal{N}, \forall t \in \mathcal{T}. \tag{7}$$

At the t-th time slot, the noise power is denoted as σ^2, and the signal-to-noise ratio (SNR) at the i-th UE side is given by

$$\gamma_i[t] = \frac{P_i^r[t] G_i[t]}{\sigma^2}, \forall i \in \mathcal{N}, \forall t \in \mathcal{T}, \tag{8}$$

where $G_i[t]$ represents the rayleigh fading model.

At the t-th time slot, if the i-th UE is in the coverage of the UAV, then the transmission rate is

$$R_i[t] = B\log_2(1+\gamma_i[t]), \forall i \in \mathcal{N}, \forall t \in \mathcal{T}, \tag{9}$$

where B represents the bandwidth. At the t-th time slot, $\xi_i[t]$ denotes the completed data transmission between the UAV and the i-th UE and can be represented as

$$\xi_i[t] = \sum_{t'=1}^{t} c_i[t'] R_i[t'], \forall i \in \mathcal{N}, \forall t \in \mathcal{T}. \tag{10}$$

Assume that the minimum communication requirements of the i-th UE is $\xi_i^{\min}$, for each round $t \in \mathcal{T}$, $\xi_i^{\min}$ is randomly generated and satisfying $\xi_i^{\min} \leq \xi^{\max}$. The UAV will communicate with the UEs in the coverage area which does not complete the communication tasks at the t-th time slot. Then, one has

$$c_i[t] = \begin{cases} 1, & \text{if } P_i^r[t] \geq P_{\min} \text{ and } \xi_i[t-1] < \xi_i^{\min}, \forall i \in \mathcal{N} \\ 0, & \text{if } P_i^r[t] < P_{\min} \text{ or } \xi_i[t-1] \geq \xi_i^{\min}, \forall i \in \mathcal{N}, \end{cases} \tag{11}$$

where $c_i[t] = 1$ denotes that the i-th UE communicates with the UAV at the t-th time slot, while $c_i[t] = 0$ indicates otherwise. The communication tasks between the UAV and all UEs can be expressed as

$$\xi_i[t_{\text{end}}] \geq \xi_i^{\min}, \forall i \in \mathcal{N}, \forall t \in \mathcal{T}, \tag{12}$$

where t_{end} represents the time slot that UAV completes all the tasks with $t_{\text{end}} \leq T_{\max}$, $T_{\max}$ is the maximal time slot which UAV can fly and its value depends on its battery capacity. The time slot duration is set as one second in the simulation.

Then, we assume that the UAV will return to the take-off site, which can be expressed as

$$\mathbf{q}[1] = \mathbf{q}[t_{\text{end}}], t_{\text{end}} \in \mathcal{T}. \tag{13}$$

The coordinates of the UAV can be given by

$$\mathbf{q}[t+1] = \mathbf{q}[t] + \begin{bmatrix} \cos(\phi[t])\cos(\psi[t]) \\ \sin(\phi[t])\cos(\psi[t]) \\ \sin(\psi[t]) \end{bmatrix} v[t], \forall t \in \mathcal{T}, \tag{14}$$

where $\phi[t]$ and $\psi[t]$ represent the heading and elevation angle of the UAV, respectively, which meet the following constraints

$$0 \leq \phi[t] \leq 2\pi, \forall t \in \mathcal{T}, \tag{15}$$

$$-\frac{\pi}{2} \leq \psi[t] \leq \frac{\pi}{2}, \forall t \in \mathcal{T}. \tag{16}$$

Also, the velocity $v[t]$ and acceleration $a[t]$ of the UAV at the t-th time slot can be represented as

$$0 \leq v[t] \leq v_{\max}, \forall t \in \mathcal{T}, \tag{17}$$

$$-a_{\max} \leq a[t] \leq a_{\max}, \forall t \in \mathcal{T}, \tag{18}$$

where $v_{\max}$ represents the maximum speed of the UAV, $a[t]$ and $a_{\max}$ represents the acceleration and the maximum acceleration of UAV, respectively. One also has

$$v[t+1] = v[t] + a[t]t, \forall t \in \mathcal{T}. \tag{19}$$

Then, the optimization goal is to minimize the flight time of the UAV to complete all tasks, which can be formulated as

$$\min_{a[t], \phi[t], \psi[t]} t_{\text{end}} \quad \text{s.t. Eqs.}(1)(2)(3)(12)(13)(15)(16)(17)(18). \tag{20}$$

According to the Eqs. (6), (9), (12), one has the following:

$$\sum_{t=1}^{t_{\text{end}}} c_i[t] \log_2(1 + \frac{(P^D[t] - L(h_i[t], s_i[t]))G_i[t]}{\sigma^2}) \geq \xi_i^{min} \tag{21}$$

Along with (20), one can see that the UAV normally applies P_{max} to communicate with the UE.

III. Proximal policy optimization 2 based method

This section introduces three important parts (state, action, and reward) of the proposed solution.

A. State

State can be defined as the vector containing all information about the environment. Here, the state s_t at the t-th time slot is denoted as

$$s_t = \left\{\mathbf{q}[t], v[t], \phi[t], \psi[t], \{\mathbf{u}_i, \xi_i[t]\}_{i \in \mathcal{N}}, N_f, t\right\}, \forall t \in \mathcal{T}, \tag{22}$$

where N_f represents the number of UEs which complete the tasks. All the above elements are normalized to [0, 1] before input to DRL. The state contains $6 + 4N + 2$ elements.

B. Action

Action space includes the possible actions that the agent may take in the environment, which can be shown as

$$a_t = \{a_a[t], \nabla\phi_a[t], \nabla\psi_a[t]\}, \forall t \in \mathcal{T}, \tag{23}$$

where $\nabla\phi_a[t]$ and $\nabla\psi_a[t]$ represent the increment of heading and elevation angle of the UAV, respectively. For the activation function of the actor network, we use $sigmoid$ for $a_a[t]$, $\nabla\phi_a[t]$ and $\nabla\psi_a[t]$. The values of the above elements can be represented as $a_a[t] \in [-1,1]$, $\nabla\phi_a[t] \in [-1,1]$, $\nabla\psi_a[t] \in [-1,1]$. Then we use the following formulas to map the above elements to the actual action space of the UAV as

$$a[t] = a_a[t]a_{\max}, \forall t \in \mathcal{T}, \tag{24}$$

$$\phi[t] = (\phi[t-1] + \nabla\phi_a[t](2\pi)) \bmod (2\pi), \forall t \in \mathcal{T}, \tag{25}$$

$$\psi[t] = (\psi[t-1] + \nabla\psi_a[t](\frac{\pi}{2})) \bmod (\frac{\pi}{2}), \forall t \in \mathcal{T}, \tag{26}$$

where $(x) \bmod (y)$ means that x takes the remainder from y.

C. Reward

The reward function consists of five parts, where the following λ_1 to λ_5 are parameters. The first part is designed as

$$r_{1,t} = \begin{cases} \lambda_1 \left(T_{\max} - t_{\text{end}}\right) & \text{if } q[1] = q\left[t_{\text{end}}\right] \text{and } N_{fc}[t] = N, \forall t \in \mathcal{T}, \\ 0 & \text{otherwise,} \end{cases} \tag{27}$$

where $r_{1,t}$ is the reward for that the UAV completing all the tasks, and $N_{fc}[t]$ denotes the number of UEs who complete the task at the t-th time slot. Since $r_{1,t}$ can be seen as the sparse reward, the probability that the UAV can obtain such a sparse reward in the training process is small. One of the more effective ways is to use reward reshaping, which can design some intermediate reward to guide the agent to obtain the final sparse reward.

Then, we further design the intermediate rewards $r_{2,t}$ to $r_{5,t}$ to speed up the training process. Thus, $r_{2,t}$ is given as

$$r_{2,t} = \lambda_2(N_{fc}[t] - N_{fc}[t-1])(T_{\max} - t), \forall t \in \mathcal{T}, \tag{28}$$

where $N_{fc}[t]$ and $N_{fc}[t-1]$ represent the number of UEs who complete tasks at the t-th and the $(t-1)$-th time slot, respectively. One can see that the less time it takes to complete the task of one UE, the greater the reward is for the UAV. Next, $r_{3,t}$ is designed as

$$r_{3,t} = \begin{cases} -\lambda_3 d_{ini}[t] & \text{if } \mathbf{q}[1] \neq \mathbf{q}[t_{\text{end}}] \text{ and } N_{fc} = N, \\ 0 & \text{otherwise ,} \end{cases} \tag{29}$$

where $d_{ini}[t]$ represents the distance between the UAV and the landing site at the t-th time slot, which is given by $d_{ini}[t] = \sqrt{||\mathbf{q}[t] - \mathbf{q}_{ini}||^2}$. One can see that the farther the UAV is from the landing site, the larger the negative reward will be. If the UAV fails to complete all UEs' communication tasks within the maximum flight time, we give the penalty as

$$r_{4,t} = \begin{cases} -\lambda_4 \sum_i^N \left(\xi_i^{\min} - \xi_i[t_{\text{end}}]\right) & \text{if } \xi_i[t_{\text{end}}] < \xi_i^{\min}, \\ 0 & \text{otherwise.} \end{cases} \tag{30}$$

Also, to prevent UAV from flying out of the working area, we add a negative reward as follows

$$r_{5,t} = \begin{cases} -\lambda_5 & \text{if the UAV flies out of the working area,} \\ 0 & \text{otherwise.} \end{cases} \tag{31}$$

Then, the total reward can be denoted as $r_t = r_{1,t} + r_{2,t} + r_{3,t} + r_{4,t} + r_{5,t}$. The purpose of the above λ_1 to λ_5 is to keep the above rewards in the same magnitude. If one reward is much larger than the others, the agent may overvalue that reward and ignore the others.

D. The Structure Of PPO2 Algorithm

The PPO2 is proposed by [10], which is based on actor-critic (AC) algorithm. The actor network, which is also called policy network, takes the state s_t as input, and outputs the action a_t. Then, the critic network takes the state s_t as input, and outputs the state value $V_w(s_t)$, where w is the parameters of the critic network. We use θ and $\pi_\theta(a_t|s_t)$ to represent the parameters and trained policy network, respectively. θ' and $\pi_{\theta'}(a_t|s_t)$ represent the parameters and sampling policy network, respectively. The loss function of policy network is

$$J(\theta) = \mathbb{E}_t[\min(e_t(\theta)\hat{A}_t, \text{clip}\left(e_t(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_t)], \tag{32}$$

where $e_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)}$ denotes the ratio of updated policy and sampling policy, ε is the clipping factor, $\hat{A}_t$ is the advantageous estimator function which can be shown as

$$\hat{A}_t = \delta_t + (\beta)\delta_{t+1} + \cdots + \cdots + (\beta)^{T-t}\delta_T, \tag{33}$$

where the β is the discount factor, and $\delta_t = r_t + \beta V_w(s_{t+1}) - V_w(s_t)$, r_t represents the reward. In addition, the loss function of the critic network is as

$$J(w) = \mathbb{E}_t\left[\delta_t^2\right], \tag{34}$$

The specific training process is shown in Algorithm 1.

In Algorithm 1, we first initialize the policy network π_θ and the critic network V_w on the main thread. Then, we create 16 parallel threads and start the training stage on each thread. On each thread, we firstly use θ and w to initialize θ_j and w_j. Then, we initialize the replay buffer $\hat{B}_j$, get s_{t_j}, and choose a_{t_j} by using the $\pi_{\theta_j}(a_{t_j}|s_{t_j})$. Then, we execute a_{t_j}, move the UAV, calculate r_{t_j}, get next s_{t_j} and store the samples to $\hat{B}_j$. If the UAV can finish task before $T_{\max}$, the data can be seen as the high quality sample, and therefore, we update the parameters θ_j and w_j for 8 times. Otherwise we only update them for once. Finally, we asynchronously update θ and w of main thread by using θ_j and w_j.

IV. NUMERICAL RESULTS

A. Simulation setting

In this section, we present the simulation settings. $(X_{\text{env}}, Y_{\text{env}}, Z_{\text{env}})$ is set as $(1000\text{m}, 1000\text{m}, 800\text{m})$ and we have 15 UEs. The take-off and landing site of UAV are $(0\text{m}, 0\text{m}, 250\text{m})$. The $T_{\max}$ is 160s; the carrier frequency is 2.5GHz, the speed of light c is $3\text{x}10^8(\text{m/s})$; the environment parameters $(\eta_a, \eta_b, \eta_{\text{LoS}}, \eta_{\text{NLoS}})$ are $(9.61, 0.16, 1, 20)$;

Algorithm 1 Pseudocode for training stage

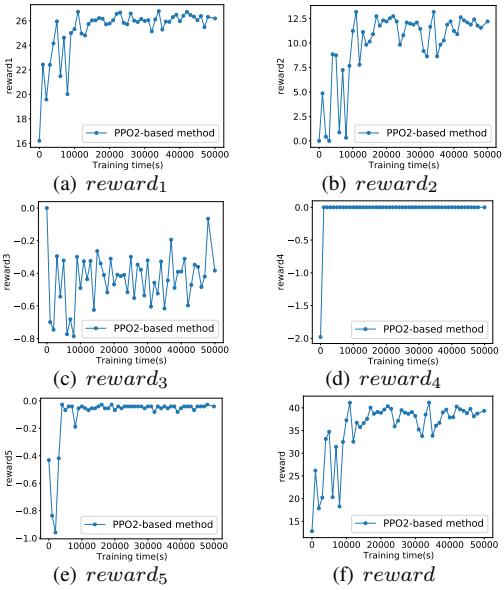
Initialize V_w, π_θ and the counter $m = 0$ are shared across different threads $thread_j, j \in \{1, ..., 16\}$;
repeat
  for $j = 1$ to 16 **do**
    Initialize $w_j \leftarrow w$, $\theta_j \leftarrow \theta$, UAV and UEs' position, replay buffer $\hat{B}_j$, then obtain state s_{t_j};
    repeat
      Choose a_{t_j} by $\pi_{\theta_j}(a_{t_j}|s_{t_j})$;
      Execute a_{t_j} based on Eqs. (24-26); and then move UAV based on Eq. (14); and then UAV selects UEs based on Eq. (11);
      Obtain the reward based on Section III. C;
      Get s_{t+1_j}; store $(s_{t_j}, a_{t_j}, r_{t_j}, s_{t+1_j})$ into $\hat{B}_j$;
      $t_j \leftarrow t_j + 1$;
    until UAV finishes all tasks or $t_j == T_{max}$
    if $t_j == T_{max}$ **then**
      Calculate $J(\theta_j)$ based on Eq. (32); update θ_j;
      Calculate $J(w_j)$ based on Eq. (34); update w_j;
    else {UAV finishs all tasks}
      Calculate $J(\theta_j)$; update θ_j;
      Calculate $J(w_j)$; update w_j;
      (Update the above for 8 times);
    end if
  end for
  Asynchronously update w and θ using w_j and θ_j;
  $m \leftarrow m + 1$;
until $m == M$

the bandwidth is 1Mhz; the limit of UAV flying altitude $z_D^{\min}$ is 250m; the $P_{\max}$ of UAV is 26dBm; the $P_{\min}$ of UEs is -70dBm; the $v_{\max}$ and $a_{\max}$ of UAV are $50(\mathrm{m/s})$ and $20(\mathrm{m/s^2})$, respectively; the $\xi^{\max}$ of all UEs is $1000KB$. The number of neurons of the actor and the critic network are $(6+4N+2, 256, 256, 4)$ and $(6+4N+2, 256, 256, 1)$, respectively. The learning rate of the actor and critic network are both 0.0001. Batch size is 512. The clipping factor ϵ and discount factor β are 0.2 and 0.98, respectively. $(\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$ are $(0.06, 0.012, 0.00005, 0.025, 0.015)$, respectively. The training epochs are 800,000. The testing group is 10,000. We use GTX4080 and i7-12700H.

B. Results and Analysis

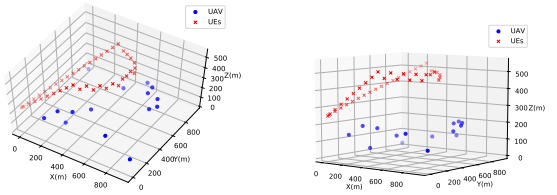
In Fig.1, we show the reward values defined in Section III. C versus the training episodes, where 16 training environments runs in parallel and the average value is recorded in the figure. We set $reward = \sum_{t=1}^{t_{\text{end}}} r_t$ and $\left\{reward_i = \sum_{t=1}^{t_{\text{end}}} r_{i,t}\right\}_{i\in\{1,...,5\}}$. One can see when the number of training in one of the training thread reaches around 30,000, the reward start to converge. During the testing stage, we randomly select one scenario and show the trajectory of UAV in Fig. 2. One can see that after training, UAV can adjust its flight altitude autonomously according to the height of the UEs, so as to achieve effecitive performance.

Next, we compare the PPO2-based algorithm with other benchmark solutions, i.e., Greedy, KMeans and exhaustive



(a) $reward_1$ (b) $reward_2$ (c) $reward_3$ (d) $reward_4$ (e) $reward_5$ (f) $reward$

Fig. 1: Reward values versus the training episodes.



(a) Overal trajectory (b) Trajectory from front view

Fig. 2: The UAV trajectory

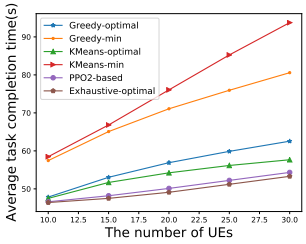


Fig. 3: Average flight time of different algorithms

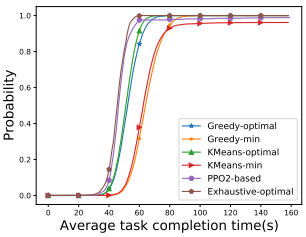


Fig. 4: Cumulative distribution function of task completion

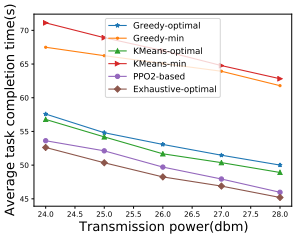


Fig. 5: Average task completion time of different algorithms under different transmission powers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI10.1109/XXXXXXXXTBC, IEEE Communications Letters

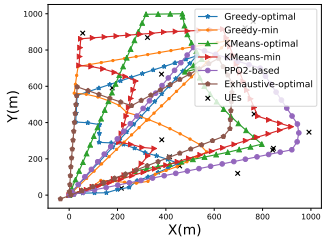


Fig. 6: 2D flight trajectory with different algorithms

search algorithm. For the Greedy algorithm, the UAV first chooses the nearest UE which does not complete the task as the next target. When the UAV completes tasks of all UEs, it flies back to the landing site in a straight line. For the KMeans algorithm, all the UEs are firstly clustered into five clusters. The UAV first flies toward the nearest cluster center until all the tasks of UEs in this cluster is completed, and then the UAV chooses the next nearest cluster. For the above algorithms, we carry out experiments at the optimal altitude and the minimum altitude of UAV, respectively. For the exhaustive search algorithm, the UAV takes all the UEs which do not complete the task as the next target and then traverses all the UEs one by one. Here, we carry out experiments at the optimal altitude. We then randomly generate 10,000 groups of UE data, then use compared algorithms to calculate the average flight time. The results are shown as Fig. 3. Firstly, one sees that the PPO2-based algorithm has better performance than the Greedy and KMeans algorithm. KMeans algorithm outperforms Greedy, as expected. Also, PPO2-based algorithm has very close performance as the exhaustive search solutions, but its time complexity and computation time are much lower than exhaustive search algorithm.

From Fig. 4, one can see PPO2-based algorithm has similar task complesion rate as the exhaustive search method. Also, we depict the average task completion time of different algorithms under different transmission powers in Fig. 5, whereas the 2-D flight trajectory of all the compared algorithms are shown in Fig. 6. From these figures, we can see that the proposed PPO2-based algorithm has considerable performance.

C. The time complexity of algorithms

For PPO2-based algorithm, we use L and l_i to denote the number of network layers and neurons in the i-th layer, and the time complexity can be given as $O(\sum_{l_i=1}^{L} l_i * l_{i+1})$. For Greedy algorithm, and for each time slot, one needs to select the nearest UE as the next flight direction, thus the time complexity is $O(N)$. For the KMeans algorithm, time complexity of clustering UEs into cluster and select the closest cluster are $O(2N_k kN)$ and $O(k)$, respectively. For the exhaustive algorithm, in each time slot, UAV can select UEs whose tasks have not been completed as the next target. The time complexity is $O(N!)$, In addition, the complexity for the common parts of the above algorithms is $O(4N)$. Then, the time complexity of the four algorithms are also shown in the Table.I.

TABLE I: The time complexity of the four algorithms

algorithms	time complexity	train time	test time
PP02-based	$O(T_{\text{end}}(\sum_{l_i=1}^{L} l_i * l_{i+1} + 4N))$	6h	0.068s
Greedy	$O(T_{\text{end}}(4N + N))$	-	0.019s
KMeans	$O(2N_k kN + T_{\text{end}}(4N + k))$	-	0.026s
Exhaustive	$O(4N * N!)$	-	$1427s$

TABLE II: The impact of UEs' position error

$\Delta\varepsilon$ (m)	average flight time	the probility of task completion rate
50	62.60s	98.07%
100	66.69s	95.03%
150	74.50s	88.03%
200	88.27s	76.07%
250	101.27s	65.07%
300	120.16s	48.73%

D. The impact of UEs' position error on PPO2-based algorithm

We show the impact of the UE position error of PPO2-based algorithm in Table II, where we add a random error value within [0, $\Delta\varepsilon$] to the UE location. It can be seen that the larger error added to the position leads to a lower mission completion rate and larger average flight time of the UAV.

V. CONCLUSION

In this paper, we designed the PPO2-based DRL algorithm to solve the trajectory design problem of UAV when the flight time is unknown. We minimized the serving time of the UAV while satisfying the QoS requirements of all the UEs. Simulation results demonstrated that the proposed framework has very good performance and low comlexity.

REFERENCES

[1] Y. Zeng, Q. Wu, and R. Zhang, "Accessing from the sky: A tutorial on uav communications for 5g and beyond," *Proceedings of the IEEE*, vol. 107, no. 12, pp. 2327–2375, 2019.

[2] X. Li, C. Zhang, R. Zhao, C. He, H. Zheng, and K. Wang, "Energy-effective offloading scheme in uav-assisted c-ran system," *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 10 821–10 832, 2022.

[3] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Trans. on Wirel. Commun.*, vol. 17, no. 3, pp. 2109–2121, 2018.

[4] R. Ding, F. Gao, and X. S. Shen, "3d uav trajectory design and frequency band allocation for energy-efficient and fair communication: A deep reinforcement learning approach," *IEEE Trans. on Wirel. Commun.*, vol. 19, no. 12, pp. 7796–7809, 2020.

[5] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and A. Nallanathan, "Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing," *IEEE Trans. Mobile Comput.*, early access,2021.

[6] Z. Wang, G. Zhang, Q. Wang, K. Wang, and K. Yang, "Completion time minimization in wireless-powered uav-assisted data collection system," *IEEE Communications Letters*, vol. 25, no. 6, pp. 1954–1958, 2021.

[7] M. Li, S. He, and 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*, vol. 9, no. 15, pp. 13 498–13 510, 2022.

[8] S. K. A. Al-Hourani and S. Lardner, "Optimal lap altitude for maximum coverage," *IEEE Wirel. Commun. Lett.*, vol. 3, no. 6, pp. 569–572, 2014.

[9] M. Alzenad, A. El-Keyi, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station for maximum coverage of users with different QoS requirements," *IEEE Wirel. Commun. Lett.*, vol. 7, no. 1, pp. 38–41, 2018.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017. [Online]. Available: https://arxiv.org/abs/1707.06347.