Deep Reinforcement Learning for Unmanned Aerial Vehicle-Assisted Vehicular Networks

Ming Zhu*, Xiao-Yang Liu*, and Xiaodong Wang

***Abstract*—Unmanned aerial vehicles (UAVs) are envisioned to complement the 5G communication infrastructure in future smart cities. Hot spots easily appear in road intersections, where effective communication among vehicles is challenging. UAVs may serve as relays with the advantages of low price, easy deployment, line-of-sight links, and flexible mobility. In this paper, we study a UAV-assisted vehicular network where the UAV jointly adjusts its transmission control (power and channel) and 3D flight to maximize the total throughput. First, we formulate a Markov decision process (MDP) problem by modeling the mobility of the UAV/vehicles and the state transitions. Secondly, we solve the target problem using a deep reinforcement learning method under unknown or unmeasurable environment variables especially in 5G, namely, the deep deterministic policy gradient (DDPG), and propose three solutions with different control objectives. Environment variables are unknown and unmeasurable, therefore, we use a deep reinforcement learning method. Moreover, considering the energy consumption of 3D flight, we extend the proposed solutions to maximize the total throughput per energy unit by encouraging or discouraging the UAV's mobility. To achieve this goal, the DDPG framework is modified. Thirdly, in a simplified model with small state space and action space, we verify the optimality of proposed algorithms. Comparing with two baseline schemes, we demonstrate the effectiveness of proposed algorithms in a realistic model.**

***Index Terms*—Unmanned aerial vehicle, vehicular networks, smart cities, Markov decision process, deep reinforcement learning, power control, channel control.**

I. Introduction

Intelligent transportation system [1] [2] [3] [4] is a key component of smart cities, which employs real-time data communication for traffic monitoring, path planning, entertainment and advertisement [5]. High speed vehicular networks [6] emerge as a key component of intelligent transportation systems that provide cooperative communications to improve data transmission performance.

With the increasing number of vehicles, the current communication infrastructure may not satisfy data transmission requirements, especially when hot spots (e.g., road intersections) appear during rush hours. Unmanned aerial vehicles (UAVs) or drones [7] can complement the 4G/5G communication infrastructure, including vehicle-to-vehicle (V2V) communications, and vehicle-to-infrastructure (V2I) communications.

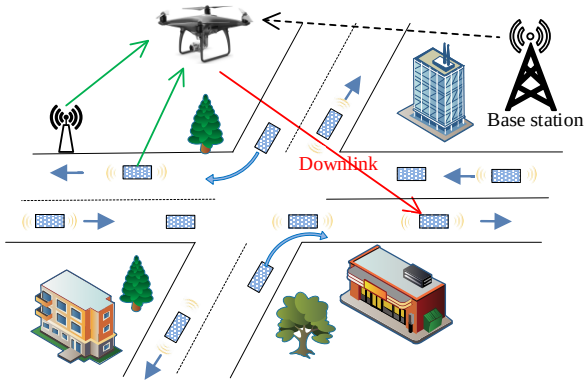

Fig. 1. The scenario of a UAV-assisted vehicular network.

Qualcomm has received a certification of authorization allowing for UAV testing below 400 feet [8]; Huawei will cooperate with China Mobile to build the first cellular test network for regional logistics UAVs [9].

A UAV-assisted vehicular network in Fig. 1 has several advantages. First, the path loss will be much lower since the UAV can move nearer to vehicles compared with stationary base stations. Secondly, the UAV is flexible in adjusting the transmission control [10] based on the mobility of vehicles. Thirdly, the quality of UAV-to-vehicle links is generally better than that of terrestrial links [11], since they are mostly line-of-sight (LoS).

Maximizing the total throughput of UAV-to-vehicle links has several challenges. First, the communication channels vary with the UAV's three-dimensional (3D) positions. Secondly, the joint adjustment of the UAV's 3D flight and transmission control (e.g., power control) cannot be solved directly using conventional optimization methods, especially when the environment variables are unknown and unmeasurable. Thirdly, the channel conditions are hard to acquire, e.g., the path loss from the UAV to vehicles is closely related to the height/density of buildings and street width.

In this paper, we propose deep reinforcement learning [12] based algorithms to maximize the total throughput of UAV-to-vehicle communications, which jointly adjusts the UAV's 3D flight and transmission control by learning through interacting with the environment. The main contributions of this paper can be summarized as follows: 1) We formulate the problem as a Markov decision process (MDP) problem to maximize the total throughput with the constraints of total transmission power and total channel; 2) We apply a deep reinforcement learning method, the deep deterministic policy gradient (DDPG), to

*Equal contribution.
M. Zhu is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China. E-mail: zhumingpassional@gmail.com.
X.-Y. Liu and X. Wang are with the Department of Electrical Engineering, Columbia University, New York, NY 10027, USA E-mail: {xiaoyang, wangx}@ee.columbia.edu.

solve the problem. DDPG is suitable to solve MDP problems with continuous states and actions. We propose three solutions with different control objectives to jointly adjust the UAV's 3D flight and transmission control. Then we extend the proposed solutions to maximize the total throughput per energy unit. To encourage or discourage the UAV's mobility, we modify the reward function and the DDPG framework; 3) We verify the optimality of proposed solutions using a simplified model with small state space and action space. Finally, we provide extensive simulation results to demonstrate the effectiveness of the proposed solutions compared with two baseline schemes.

The remainder of the paper is organized as follows. Section II discusses related works. Section III presents system models and problem formulation. Solutions are proposed in Section IV. Section V presents the performance evaluation. Section VI concludes this paper.

II. Related Works

The dynamic control for the UAV-assisted vehicular networks includes flight control and transmission control. Flight control mainly includes the planning of flight path, time, and direction. Yang *et al.* [13] proposed a joint genetic algorithm and ant colony optimization method to obtain the best UAV flight paths to collect sensory data in wireless sensor networks. To further minimize the UAVs' travel duration under certain constraints (e.g., energy limitations, fairness, and collision), Garraffa *et al.* [14] proposed a two-dimensional (2D) path planning method based on a column generation approach. Liu *et al.* [15] proposed a deep reinforcement learning approach to control a group of UAVs by optimizing the flying directions and distances to achieve the best communication coverage in the long run with limited energy consumption.

The transmission control of UAVs mainly concerns resource allocations, e.g., access selection, transmission power and bandwidth/channel allocation. Wang *et al.* [16] presented a power allocation strategy for UAVs considering communications, caching, and energy transfer. In a UAV-assisted communication network, Yan *et al.* [17] studied a UAV access selection and base station bandwidth allocation problem, where the interaction among UAVs and base stations was modeled as a Stackelberg game, and the uniqueness of a Nash equilibrium was obtained.

Joint control of both UAVs' flight and transmission has also be considered. Wu *et al.* [18] considered maximizing the minimum achievable rates from a UAV to ground users by jointly optimizing the UAV's 2D trajectory and power allocation. Zeng *et al.* [19] proposed a convex optimization method to optimize the UAV's 2D trajectory to minimize its mission completion time while ensuring each ground terminal recovers the file with high probability when the UAV disseminates a common file to them. Zhang *et al.* [20] considered the UAV mission completion time minimization by optimizing its 2D trajectory with a constraint on the connectivity quality from base stations to the UAV. However, most existing research works neglected adjusting UAVs' height to obtain better quality of links by avoiding various obstructions or non-line-of-sight (NLoS) links.

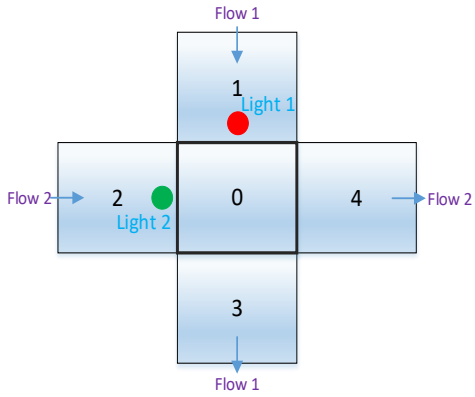

Fig. 2. A one-way-two-flow road intersection.

Fan *et al.* [21] optimized the UAV's 3D flight and transmission control together; however, the 3D position optimization was converted to a 2D position optimization by the LoS link requirement. The existing deep reinforcement learning based methodd only handle UAVs' 2D flight and simple transmission control decisions. For example, Challita *et al.* [22] proposed a deep reinforcement learning based method for a cellular UAV network by optimizing the 2D path and cell association to achieve a tradeoff between maximizing energy efficiency and minimizing both wireless latency and the interference on the path. A similar scheme is applied to provide intelligent traffic light control in [23].

In addition, most existing works assumed that the ground terminals are stationary; whereas in reality, some ground terminals move with certain patterns, e.g., vehicles move under the control of traffic lights. This work studies a UAV-assisted vehicular network where the UAV's 3D flight and transmission control can be jointly adjusted, considering the mobility of vehicles in a road intersection.

III. System Models and Problem Formulation

In this section, we first describe the traffic model and communication model, and then formulate the target problem as a Markov decision process. The variables in the communication model are listed in Table I for easy reference.

A. Traffic Model

We start with a one-way-two-flow road intersection, as shown in Fig. 2, while a much more complicated scenario in Fig. 7 will be described in Section V-B. Five blocks are numbered as 0, 1, 2, 3, and 4, where block 0 is the intersection. We assume that each block contains at most one vehicle, indicated by binary variables $\boldsymbol{n} = (n^0, ..., n^4) \in \{0, 1\}$. There are two traffic flows in Fig. 2,

- "Flow 1": $1 \rightarrow 0 \rightarrow 3$;
- "Flow 2": $2 \rightarrow 0 \rightarrow 4$.

The traffic light L has four configurations:

- $L = 0$: red light for flow 1 and green light for flow 2;
- $L = 1$: red light for flow 1 and yellow light for flow 2;
- $L = 2$: green light for flow 1 and red light for flow 2;

TABLE I
VARIABLES IN COMMUNICATION MODEL

h_t^i, H_t^i	channel power gain and channel state from the UAV to a vehicle in block i in time slot t.
ψ_t^i	SINR from the UAV to a vehicle in block i in time slot t.
d_t^i, D_t^i	horizontal distance and Euclidean distance between the UAV and a vehicle in block i.
P, C, b	total transmission power, total number of channels, and bandwidth of each channel.
ρ_t^i, c_t^i	transmission power and number of channels allocated for the vehicle in block i in time slot t.

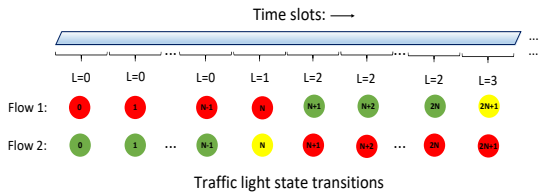

Fig. 3. Traffic light states along time.

- $L=3$: yellow light for flow 1 and red light for flow 2.

Time is partitioned into slots with equal duration. The duration of a green or red light occupies N time slots, and the duration of a yellow light occupies a time slot, which are shown in Fig. 3. We assume that each vehicle moves one block in a time slot if the traffic light is green.

B. Communication Model

We focus on the downlink communications (UAV-to-vehicle), since they are directly controlled by the UAV. There are two channel states of each UAV-to-vehicle link, line-of-sight (LoS) and non-line-of-sight (NLoS). Let x and z denote the block (horizontal position) and height of the UAV respectively, where $x \in \{0,1,2,3,4\}$ corresponds to these five blocks in Fig. 2, and z is discretized to multiple values. We assume that the UAV stays above the five blocks since the UAV trends to get nearer to vehicles. Next, we describe the communication model, including the channel power gain, the signal to interference and noise ratio (SINR), and the total throughput.

First, the channel power gain between the UAV and a vehicle in block i in time slot t is h_t^i with a channel state $H_t^i \in \{\text{NLoS}, \text{LoS}\}$. h^i is formulated as [10] [24]

$$h_t^i = \begin{cases} (D_t^i)^{-\beta_1}, & \text{if } H_t^i = \text{LoS}, \\ \beta_2 (D_t^i)^{-\beta_1}, & \text{if } H_t^i = \text{NLoS}, \end{cases} \tag{1}$$

where D_t^i is the Euclidean distance between the UAV and the vehicle in block i in time slot t, β_1 is the path loss exponent, and β_2 is an additional attenuation factor caused by NLoS connections.

The probabilities of LoS and NLoS links between the UAV and a vehicle in block i in time slot t are [25]

$$p(H_t^i = \text{LoS}) = \frac{1}{1 + \alpha_1 \exp(-\alpha_2 (\frac{180}{\pi} \arctan \frac{z}{d_t^i} - \alpha_1))}, \tag{2}$$

$$p(H_t^i = \text{NLoS}) = 1 - p(H_t^i = \text{LoS}), \ i \in \{0,1,2,3,4\}, \tag{3}$$

where α_1 and α_2 are system parameters depending on the environment (height/density of buildings, and street width, etc.). We assume that α_1, α_2, β_1, and β_2 have fixed values among all blocks in an intersection. d_t^i is the horizontal distance in time slot t. The angle $\frac{180}{\pi} \arctan \frac{z}{d_t^i}$ is measured in "degrees" with the range $0^\circ \sim 90^\circ$. Both d_t^i and z_t are discrete variables, therefore, $D_t^i = \sqrt{(d_t^i)^2 + z_t^2}$ is also a discrete variable.

Secondly, the SINR ψ_t^i in time slot t from the UAV to a vehicle in block i is characterized as [26]

$$\psi_t^i = \frac{\rho_t^i h_t^i}{b c_t^i \sigma^2}, \ i \in \{0,1,2,3,4\}, \tag{4}$$

where b is the equal bandwidth of each channel, ρ_t^i and c_t^i are the allocated transmission power and number of channels for the vehicle in block i in time slot t, respectively, and σ^2 is the additive white Gaussian noise (AWGN) power spectrum density, and h^i is formulated by (1). We assume that the UAV employs orthogonal frequency division multiple access (OFDMA) [27]; therefore, there is no interference among these channels.

Thirdly, the total throughput (reward) of UAV-to-vehicle links is formulated as [28]

$$\sum_{i \in \{0,1,2,3,4\}} b c_t^i \log(1+\psi_t^i) = \sum_{i \in \{0,1,2,3,4\}} b c_t^i \log(1+\frac{\rho_t^i h_t^i}{b c_t^i \sigma^2}). \tag{5}$$

C. MDP Formulation

The UAV aims to maximize the total throughput with the constraints of total transmission power and total channels:

$$\sum_{i \in \{0,1,2,3,4\}} \rho_t^i \le P, \quad \sum_{i \in \{0,1,2,3,4\}} c_t^i \le C,$$
$$0 \le \rho_t^i \le \rho_{\max}, \quad 0 \le c_t^i \le c_{\max}, \ i \in \{0,1,2,3,4\},$$

where P is the total transmission power, C is the total number of channels, $\rho_{\max}$ is the maximum power allocated to a vehicle, $c_{\max}$ is the maximum number of channels allocated to a vehicle, ρ_t^i is a discrete variable, and c_t^i is a nonnegative integer variable.

The UAV-assisted communication is modeled as a Markov decision process (MDP). On one hand, from (2) and (3), we know that the channel state of UAV-to-vehicle links follows a stochastic process. On the other hand, the arrival of vehicles follows a stochastic process under the control of the traffic light, e.g., (12) and (13).

Under the MDP framework, the state space $\mathcal{S}$, action space $\mathcal{A}$, reward r, policy π, and state transition probability $p(s_{t+1}|s_t, a_t)$ of our problem are defined as follows.

- State $\mathcal{S} = (L, x, z, \boldsymbol{n}, \boldsymbol{H})$, where L is the traffic light state, (x, z) is the UAV's 3D position with $x \in \{0,1,2,3,4\}$ being the block and z being the height, and $\boldsymbol{H} = (H^0, ..., H^4)$ is the channel state from the UAV to each block $i \in \{0,1,2,3,4\}$ with $H^i \in \{\text{NLoS}, \text{LoS}\}$. Let $z \in [z_{\min}, z_{\max}]$, where $z_{\min}$ and $z_{\max}$ are the UAV's minimum and maximum height, respectively. The block x is the location projected from UAV's 3D position to the road.

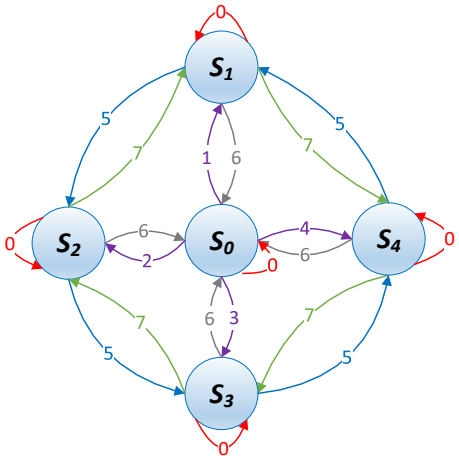

Fig. 4. The position state transition diagram when the UAV's height is fixed.

- Action $\mathcal{A} = (\boldsymbol{f}, \boldsymbol{\rho}, \boldsymbol{c})$ denotes the action set. f^x denotes the horizontal flight, and f^z denotes the vertical flight, both of which constitute the UAV's 3D flight $\boldsymbol{f} = (f^x, f^z)$. With respect to horizontal flight, we assume that the UAV can hover or flight to its adjacent block in a time slot, thus $f^x \in \{0, 1, ..., 7\}$ in Fig. 4. With respect to vertical flight, we assume

$$f^z \in \{-5, 0, 5\}, \tag{6}$$

which means that the UAV can flight downward 5 meters, horizontally, and up 5 meters in a time slot. The UAV's height changes as

$$z_{t+1} = f_t^z + z_t. \tag{7}$$

$\boldsymbol{\rho} = (\rho_t^0, ..., \rho_t^4)$ and $\boldsymbol{c} = (c_t^0, ..., c_t^4)$ are the transmission power and channel allocation actions for those five blocks, respectively. At the end of time slot t, the UAV moves to a new 3D position according to action $\boldsymbol{f}$, and over time slot t, the transmission power and number of channels are $\boldsymbol{\rho}$ and $\boldsymbol{c}$, respectively.

- Reward $r(s_t, a_t) = \sum_{i\in\{0,1,2,3,4\}} bn_t^i c_t^i \log(1 + \frac{\rho_t^i h_t^i}{bc_t^i \sigma^2})$ is the total throughput after a transition from state s_t to s_{t+1} taking action a_t. Note that the total throughput over the t-th time slot is measured at the state $s_t = (L_t, x_t, z_t, \boldsymbol{n}_t, \boldsymbol{H}_t)$.
- Policy π is the strategy for the UAV, which maps states to a probability distribution over the actions $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, where $\mathcal{P}(\cdot)$ denotes probability distribution. In time slot t, the UAV's state is $s_t = (L_t, x_t, z_t, \boldsymbol{n}_t, \boldsymbol{H}_t)$, and its policy π_t outputs the probability distribution over the action a_t. We see that the policy indicates the action preference of the UAV.
- State transition probability $p(s_{t+1}|s_t, a_t)$ formulated in (8) is the probability of the UAV entering the new state s_{t+1}, after taking the action a_t at the current state s_t. At the current state $s_t = (L_t, x_t, z_t, \boldsymbol{n}_t, \boldsymbol{H}_t)$, after taking the 3D flight and transmission control $a_t = (\boldsymbol{f}, \boldsymbol{\rho}, \boldsymbol{c})$, the UAV moves to the new 3D position (x_{t+1}, z_{t+1}), and the channel state changes to $\boldsymbol{H}_{t+1}$, with the traffic light changes to L_{t+1} and the number of vehicles in each block changes to $\boldsymbol{n}_{t+1}$.

The state transitions of the traffic light along time are shown in Fig. 3. The transition of the channel state for UAV-to-vehicle links is a stochastic process, which is reflected by (2) and (3).

Next, we discuss the MDP in three aspects: the state transition probability, the state transitions of the number of vehicles in each block, and the UAV's 3D position. Note that the transmission power control and channel control do not affect the traffic light, the channel state, the number of vehicles, and the UAV's 3D position.

First, we discuss the state transition probability $p(s_{t+1}|s_t, a_t) = p((L_{t+1}, x_{t+1}, z_{t+1}, \boldsymbol{n}_{t+1}, \boldsymbol{H}_{t+1}) |(L_t, x_t, z_t, \boldsymbol{n}_t, \boldsymbol{H}_t), (\boldsymbol{f}_t, \boldsymbol{\rho}_t, \boldsymbol{c}_t))$. The UAV's 3D fight only affects the UAV's 3D position state and the channel state, the traffic light state of the next time slot relies on the current traffic light state, and the number of vehicles in each block of the next time slot relies on the current number of vehicles and the traffic light state. Therefore, the state transition probability is

$$\begin{aligned} p(s_{t+1}|s_t, a_t) = {} & p(x_{t+1}, z_{t+1}|x_t, z_t, \boldsymbol{f}_t) \\ & \times p(\boldsymbol{H}_{t+1}|x_t, z_t, \boldsymbol{f}_t) \times p(L_{t+1}|L_t) \\ & \times p(\boldsymbol{n}_{t+1}|L_t, \boldsymbol{n}_t), \end{aligned} \tag{8}$$

where $p(x_{t+1}, z_{t+1}|x_t, z_t, \boldsymbol{f}_t)$ is easily obtained by the 3D position state transition based on the UAV's flight actions in Fig. 4, $p(\boldsymbol{H}_{t+1}|x_t, z_t, \boldsymbol{f}_t)$ is easily obtained by (2) and (3), $p(L_{t+1}|L_t)$ is obtained by the traffic light state transition in Fig. 3, and $p(\boldsymbol{n}_{t+1}|L_t, \boldsymbol{n}_t)$ is easily obtained by (9) $\sim$ (13).

Secondly, we discuss the state transitions of the number of vehicles in each block. It is a stochastic process. The UAV's states and actions do not affect the number of vehicles of all blocks. Let λ_1 and λ_2 be the probabilities of the arrivals of new vehicles in flow 1 and 2, respectively.

The state transitions for the number of vehicles in block 0, 3, and 4 are

$$n_{t+1}^0 = \begin{cases} n_t^2, & \text{if } L_t = 0, \\ n_t^1, & \text{if } L_t = 2, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

$$n_{t+1}^3 = \begin{cases} n_t^0, & \text{if } L_t = 2, 3, \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$

$$n_{t+1}^4 = \begin{cases} n_t^0, & \text{if } L_t = 0, 1, \\ 0, & \text{otherwise}. \end{cases} \tag{11}$$

The transition probability is 1 in (9), (10) and (11) since the transitions are deterministic in block 0, 3, and 4. While the state transition probabilities for the number of vehicles in block 1 and 2 are nondeterministic, moreover, both of them are affected by their current number of vehicles and the traffic light. Taking block 1 when the traffic light state $L_t = 2$ as an example, the probability for the number of vehicles is

$$p(n_{t+1}^1 = 1|L_t = 2) = \lambda_1, \tag{12}$$

$$p(n_{t+1}^1 = 0|L_t = 2) = 1 - \lambda_1. \tag{13}$$

When $(n_t^1 = 0, L_t \neq 2)$ and $(n_t^1 = 1, L_t \neq 2)$, the probability for the number of vehicles will be obtained in a similar way.

Thirdly, we discuss the state transition of the UAV's 3D position. It includes block transitions and height transitions. The UAV's height transition is formulated in (7). If the UAV's height is fixed, the corresponding position state transition diagram is shown in Fig. 4, where $\{S_i\}_{i\in\{0,1,2,3,4\}}$ denotes the block of the UAV: 0 denotes staying in the current block; $\{1, 2, 3, 4\}$ denotes a flight from block 0 to the other blocks (1, 2, 3, and 4); 5 denotes an anticlockwise flight; 6 denotes a flight from block 1, 2, 3, or 4 to block 0; 7 denotes a clockwise flight.

IV. Proposed Solutions

In this section, we first describe the motivation, and then present an overview of Q-learning and the deep deterministic policy gradient algorithm, and then propose solutions with different control objectives, and finally present an extension of solutions that takes into account the energy consumption of 3D flight.

A. Motivation

Deep reinforcement learning methods are suitable for the target problem since environment variables are unknown and unmeasurable. For example, α_1 and α_2 are affected by the height/density of buildings, and the height and size of vehicles, etc. β_1 and β_2 are time dependent and are affected by the current environment such as weather [29]. Although UAVs can detect the LoS/NloS links using equipped cameras, it is very challenging to detect them accurately for several reasons. First, the locations of recievers on vehicles should be labeled for detection. Secondly, it is hard to detect receivers accurately using computer vision technology since receivers are much small than vehicles. Thirdly, it requires automobile manufacturers to label the loactions of receivers, which may not be satisfied in several years. Therefore, it requires a large amount of labor to test these environment variables accurately.

It is hard to obtain the optimal strategies even all environment variables are known. Existing works [30] [31] obtain the near-optimal strategies in the 2D flight scenario when users are stationary, however, they are not capable of solving our target problem since the UAV adjusts its 3D position and vehicles move with their patterns under the control of traffic lights.

B. Q-learning

The state transition probabilities of MDP are unknown in our problem, since some variables are unknown, e.g., α_1, α_2, λ_1, and λ_2. Our problem cannot be solved directly using conventional MDP solutions, e.g., dynamic programming algorithms, policy iteration and value iteration algorithms. Therefore, we apply the reinforcement learning (RL) approach. The return from a state is defined as the sum of discounted future reward $\sum_{i=t}^{T}\gamma^{i-t}r(s_i, a_i)$, where T is the total number of time slots, and $\gamma \in (0, 1)$ is a discount factor that diminishes the future reward and ensures that the sum of an infinite number of rewards is still finite. Let $Q^{\pi}(s_t, a_t) = \mathbb{E}_{a_i\sim\pi}[\sum_{i=t}^{T}\gamma^{i-t}r(s_i, a_i)|s_t, a_t]$ represents the expected return after taking action a_t in state s_t under policy π. The Bellman equation gives the optimality condition in conventional MDP solutions [32]:

$$Q^{\pi}(s_t, a_t) = \sum_{s_{t+1}, r_t} p(s_{t+1}, r_t|s_t, a_t)\left[r_t + \gamma \max_{a_{t+1}} Q^{\pi}(s_{t+1}, a_{t+1})\right].$$

Q-learning [33] is a classical model-free RL algorithm [34]. Q-learning with the essence of exploration and exploitation aims to maximize the expected return by interacting with the environment. The update of $Q(s_t, a_t)$ is

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)], \quad (14)$$

where α is a learning rate.

Q-learning uses the ϵ-greedy strategy [35] to select an action, so that the agent behaves greedily most of the time, but selects randomly among all the actions with a small probability ϵ. The ϵ-greedy strategy is defined as follows

$$a_t = \begin{cases} \arg\max_a Q(s_t, a), & \text{with probability } 1-\epsilon, \\ \text{a random action}, & \text{with probability } \epsilon. \end{cases} \quad (15)$$

The Q-learning algorithm [32] is shown in Alg. 1. Line 1 is initialization. In each episode, the inner loop is executed in lines $4 \sim 7$. Line 5 selects an action using (15), and then the action is executed in line 6. Line 7 updates the Q-value.

Q-learning cannot solve our problem because of several limitations. 1) Q-learning can only solve MDP problems with small state space and action space. However, the state space and action space of our problem are very large. 2) Q-learning cannot handle continuous state or action space. The UAV's transmission power allocation actions are continuous. The transmission power control is a continuous action in reality. If we discretize the transmission power allocation actions, and use Q-learning to solve it, the result may be far from the optimum. 3) Q-learning will converge slowly using too many computational resources [32], and this is not practical in our problem. Therefore, we adopt the deep deterministic policy gradient algorithm to solve our problem.

C. Deep Deterministic Policy Gradient

The deep deterministic policy gradient (DDPG) method [36] uses deep neural networks to approximate both action policy π and value function $Q(s, a)$. This method has two advantages: 1) it uses neural networks as approximators, essentially compressing the state and action space to much smaller latent parameter space, and 2) the gradient descent method can be used to update the network weights, which greatly speeds up the convergence and reduces the computational time. Therefore, the memory and computational resources are largely saved. In real systems, DDPG exploits the powerful skills introduced in AlphaGo zero [37] and Atari game playing [38], including experience replay buffer, actor-critic approach, soft update, and exploration noise.

1) Experience replay buffer R_b stores transitions that will be used to update network parameters. At each time slot t,

Algorithm 1: Q-learning-based algorithm

Input: the number of episodes K, the learning rate α, parameter ϵ.
1: Initialize all states. Initialize $Q(s,a)$ for all state-action pairs randomly.
2: **for** episode $k=1$ to K
3: Observe the initial state s_1.
4: **for** each slot $t=1$ to T
5: Select the UAV's action a_t from state s_t using (15).
6: Execute the UAV's action a_t, receive reward r_t, and observe a new state s_{t+1} from the environment.
7: Update Q-value function: $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right]$.

a transition (s_t, a_t, r_t, s_{t+1}) is stored in R_b. After a certain number of time slots, each iteration samples a mini-batch of $M = |\Omega|$ transitions $\{(s^j, a^j, r^j, s^j)\}_{j\in\Omega}$ to train neural networks, where Ω is a set of indices of sampled transitions from R_b. "Experience replay buffer" has two advantages: 1) enabling the stochastic gradient decent method [39]; and 2) removing the correlations between consecutive transitions.

2) Actor-critic approach: the critic approximates the Q-value, and the actor approximates the action policy. The critic has two neural networks: the online Q-network Q with parameter θ^Q and the target Q-network Q' with parameter $\theta^{Q'}$. The actor has two neural networks: the online policy network μ with parameter θ^μ and the target policy network μ' with parameter $\theta^{\mu'}$. The training of these four neural networks are discussed in the next subsection.

3) Soft update with a low learning rate $\tau \ll 1$ is introduced to improve the stability of learning. The soft updates of the target Q-network Q' and the target policy network μ' are as follows

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} = \theta^{Q'} + \tau(\theta^Q - \theta^{Q'}), \quad (16)$$

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'} = \theta^{\mu'} + \tau(\theta^\mu - \theta^{\mu'}). \quad (17)$$

4) Exploration noise is added to the actor's target policy to output a new action

$$a_t = \mu'(s_t|\theta^{\mu'}) + \mathcal{N}_t. \quad (18)$$

There is a tradeoff between exploration and exploitation, and the exploration is independent from the learning process. Adding exploration noise in (18) ensures that the UAV has a certain probability of exploring new actions besides the one predicted by the current policy $\mu'(s_t|\theta^{\mu'})$, and avoids that the UAV is trapped in a local optimum.

D. Deep Reinforcement Learning-based Solutions

The UAV has two transmission controls, power and channel. We use the power allocation as the main control objective for two reasons. 1) Once the power allocation is determined, the channel allocation will be easily obtained in OFDMA. According to Theorem 4 of [40], in OFDMA, if all links have the equal weights just as our reward function (5), the transmitter should send messages to the receiver with the strongest channel in each time slot. In our problem, the strongest channel is not determined since the channel state (LoS or NLoS) is a random process. DDPG trends to allocate more power to the strongest channels with large probabilities, therefore, channel allocation will be easily obtained based on power allocation actions. 2) Power allocation is continuous, and DDPG is suitable to handle these actions. However, if we use DDPG for the channel allocation, the number of action variables will be very large and the convergence will be very slow, since the channel allocation is discrete and the number of channels is generally large (e.g., 200) especially in rush hours. We choose power control or flight as control objectives since controlling power and flight is more efficient than controlling channel. Moreover, the best channel allocation strategy can be obtained indirectly if the power is allocated in OFDMA. Based on the above analysis, we propose three algorithms:

- PowerControl: the UAV adjusts the transmission power allocation using the actor network at a fixed 3D position, and the channels are allocated to vehicles by Alg.2 in each time slot.
- FlightControl: the UAV adjusts its 3D flight using the actor network, and the transmission power and channel allocation are equally allocated to each vehicle in each time slot.
- JointControl: the UAV adjusts its 3D flight and the transmission power allocation using the actor network, and the channels are allocated to vehicles by Alg.2 in each time slot.

To allocate channels among blocks, we introduce a variable denoting the average allocated power of a vehicle in block i:

$$\bar{\rho}_t^i = \begin{cases} \frac{\rho_t^i}{n_t^i}, & \text{if } n_t^i \neq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (19)$$

The channel allocation algorithm is shown in Alg. 2, which is executed after obtaining the power allocation actions. As the above description, it achieves the best channel allocation in OFDMA if the power allocation is known [40]. Line 1 is the initialization. Lines 2 $\sim$ 3 calculate and sort $\bar{\boldsymbol{\rho}}_t = \{\bar{\rho}_t^i\}_{i\in\{0,1,2,3,4\}}$. Line 5 assigns the maximum number of channels to the current possibly strongest channel, and line 6 updates the remaining total number of channels.

The DDPG-based algorithms are given in Alg. 3. The algorithm has two parts: initializations, and the main process. First, we describe the initializations in lines 1 $\sim$ 3. In line 1, all states are initialized: the traffic light L is initialized as 0, the number of vehicles $\boldsymbol{n}$ in all blocks is 0, the UAV's block and height are randomized, and the channel state H^i for each block i is set as LoS or NLoS with the same probability. Note that the action space DDPG controls in PowerControl, FlightControl, and JointControl is different. Line 2 initializes the parameters

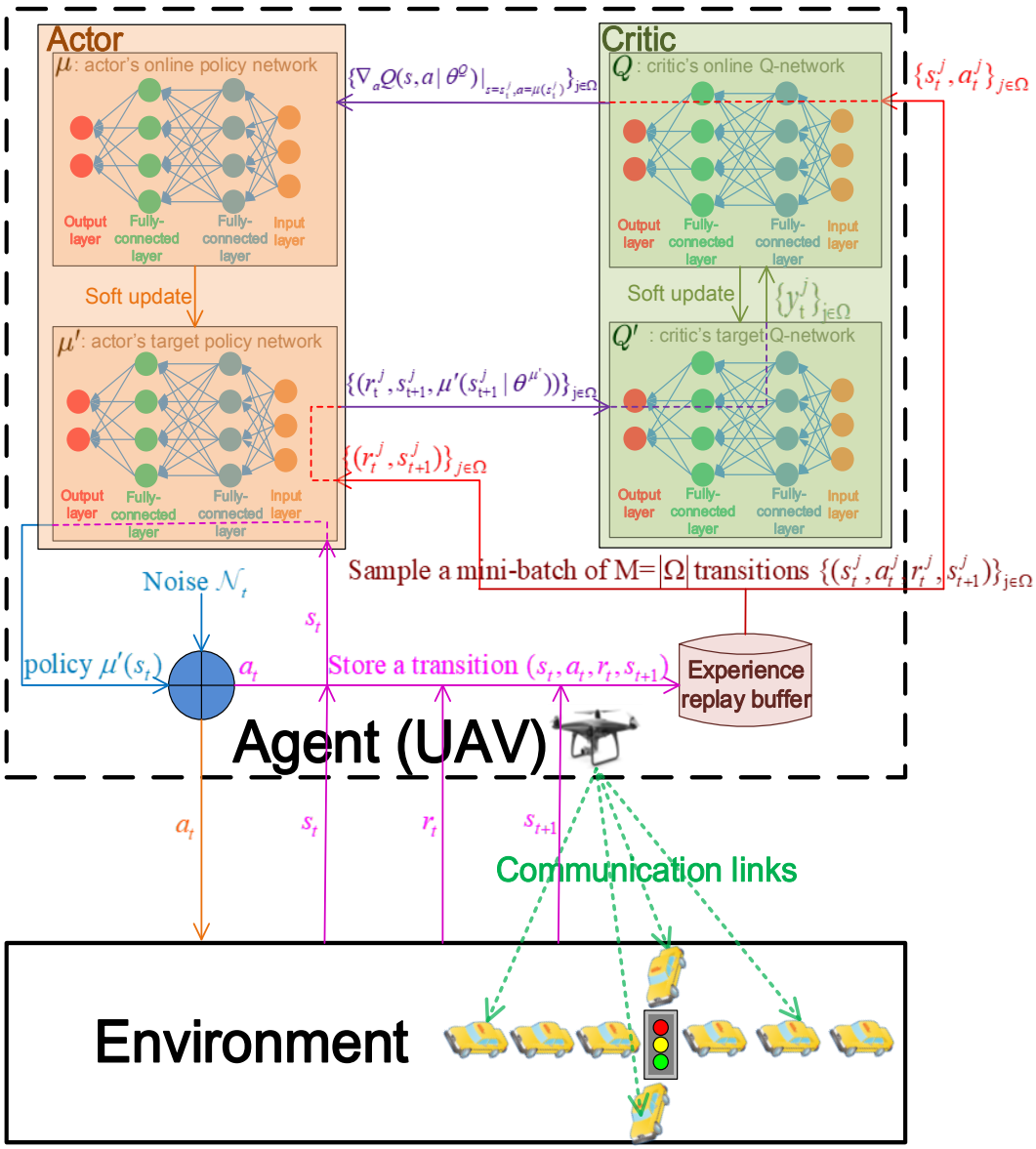

Fig. 5. Framework of the DDPG algorithm.

Algorithm 2: Channel allocation in time slot t

Input: the power allocation $\boldsymbol{\rho}$, the number of vehicles in all blocks $\boldsymbol{n}$, the maximum number of channels allocated to a vehicle $c_{\max}$, the total number of channels C.

Output: the channel allocation $\boldsymbol{c}_t$ for all blocks.

1: Initialize the remaining total number of channels $C_r \leftarrow C$.
2: Calculate the average allocated power for each vehicle in all blocks $\bar{\boldsymbol{\rho}}_t$ by (19).
3: Sort $\bar{\boldsymbol{\rho}}_t$ by the descending order, and obtain a sequence of block indices $\boldsymbol{J}$.
4: **for** block $j \in \boldsymbol{J}$
5: $c_t^j \leftarrow \min(C_r, n_t^j c_{\max})$.
7: $C_r \leftarrow C_r - c_t^j$.
8: Return $\boldsymbol{c}_t$.

Algorithm 3: DDPG-based algorithms: PowerControl, FlightControl, and JointControl

Input: the number of episodes K, the number of time slots T in an episode, the mini-batch size M, the learning rate τ.

1: Initialize all states, including the traffic light state L, the UAV's 3D position (x, z), the number of vehicles $\boldsymbol{n}$ and the channel state $\boldsymbol{H}$ in all blocks.
2: Randomly initialize critic's online Q-network parameters θ^Q and actor's online policy network parameters θ^μ, and initialize the critic's target Q-network parameters $\theta^{Q'} \leftarrow \theta^Q$ and actor's target policy network parameters $\theta^{\mu'} \leftarrow \theta^\mu$.
3: Allocate an experience replay buffer R_b.
4: **for** episode $k = 1$ to K
5: Initialize a random process (a standard normal distribution) $\mathcal{N}$ for the UAV's action exploration.
6: Observe the initial state s_1.
7: **for** $t = 1$ to T
8: Select the UAV's action $\bar{a}_t = \mu'(s_t|\theta^{\mu'}) + \mathcal{N}_t$ according to the policy of μ' and the exploration noise $\mathcal{N}_t$.
9: **if** PowerControl
10: Combine the channel allocation in Alg. 2 and $\bar{a}_t$ as the UAV's action a_t at a fixed 3D position.
11: **if** FlightControl
12: Combine the equal transmission power, equal channel allocation and $\bar{a}_t$ (3D flight) as the UAV's action a_t.
13: **if** JointControl
14: Combine the 3D flight action, the channel allocation in Alg. 2 and $\bar{a}_t$ as the UAV's action a_t.
15: Execute the UAV's action a_t, and receive reward r_t, and observe new state s_{t+1} from the environment.
16: Store transition (s_t, a_t, r_t, s_{t+1}) in the UAV's experience replay buffer R_b.
17: Sample R_b to obtain a random mini-batch of M transitions $\{(s_t^j, a_t^j, r_t^j, s_{t+1}^j)\}_{j\in\Omega} \subseteq R_b$, where Ω is a set of indices of sampled transitions with $|\Omega| = M$.
18: The critic's target Q-network Q' calculates and outputs $y_t^j = r_t^j + \gamma Q'(s_{t+1}^j, \mu'(s_{t+1}^j|\theta^{\mu'})|\theta^{Q'})$ to the critic's online Q-network Q.
19: Update the critic's online Q-network Q to make its Q-value fit y_t^j by minimizing the loss function: $\nabla_{\theta^Q}\text{Loss}_t(\theta^Q) = \nabla_{\theta^Q}[\frac{1}{M}\sum_{j=1}^{M}(y_t^j - Q(s_t^j, a_t^j|\theta^Q))^2]$.
20: Update the actor's online policy network μ based on the input $\{\nabla_a Q(s, a|\theta^Q)|_{s=s_t^j, a=\mu(s_t^j)}\}_{j\in\Omega}$ from Q using the policy gradient by the chain rule: $\frac{1}{M}\sum_{j\in\Omega}\mathbb{E}_{s_t}[\nabla_a Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t)}\nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s=s_t}]$.
21: Soft update the critic's target Q-network Q' and actor's target policy network μ' to make the evaluation of the UAV's actions and the UAV's policy more stable: $\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$, $\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$.

of the critic and actor. Line 3 allocates an experience replay buffer R_b.

Secondly, we present the main process. Line 5 initializes a random process for action exploration. Line 6 receives an initial state s_1. Let $\bar{a}_t$ be the action DDPG controls, and a_t be the UAV's all action. Line 8 selects an action according to $\bar{a}_t$ and an exploration noise $\mathcal{N}_t$. Lines 9 $\sim$ 10 combine the channel allocation actions in Alg. 2 and $\bar{a}_t$ as a_t at a fixed 3D position in PowerControl. Lines 11 $\sim$ 12 combine the equal transmission power, equal channel allocation actions and $\bar{a}_t$ (3D flight) as a_t in FlightControl. Lines 13 $\sim$ 14 combine the 3D flight action, the channel allocation actions in Alg. 2 and $\bar{a}_t$ as a_t in JointControl. Line 15 executes the UAV's action a_t, and then the UAV receives a reward and all states are updated. Line 16 stores a transition into R_b. In line 17, a random mini-batch of transitions are sampled from R_b. Line 18 sets the value of y^j for the critic's online Q-network. Lines 19 $\sim$ 21 update all network parameters.

The DDPG-based algorithms in Alg. 3 in essence are the approximated Q-learning method in Alg. 1. The exploration noise in line 8 approximates the second case of (15) in Q-learning. Lines 18 $\sim$ 19 in Alg. 3 make $\left[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]$ in line 7 of Alg. 1 converge. Line 20 of Alg. 3 approximates the first case of (15) in Q-learning, since both of them aim to obtain the policy of the maximum Q-value. The soft update of Q' in line 21 of Alg. 3 is exactly (14) in Q-learning, where τ and α are learning rates. Next, we discuss the training and test stages of proposed solutions.

1) In the training stage, we train the actor and the critic, and store the parameters of their neural networks. Fig. 5 illustrates the data flow and parameter update process. The training stage has two parts. First, Q and μ are trained through a random mini-batch of transitions sampled from the experience replay buffer R_b. Secondly, Q' and μ' are trained through soft update.

The training process is as follows. A mini-batch of M transitions $\{(s_t^j, a_t^j, r_t^j, s_{t+1}^j)\}_{j\in\Omega}$ are sampled from R_b, where Ω is a set of indices of sampled transitions from R_b with $|\Omega| = M$. Then two data flows are outputted from R_b: $\{r_t^j, s_{t+1}^j\}_{j\in\Omega} \rightarrow \mu'$, and $\{s_t^j, a_t^j\}_{j\in\Omega} \rightarrow Q$. μ' outputs $\{r_t^j, s_{t+1}^j, \mu'(s_{t+1}^j|\theta^{\mu'})\}_{j\in\Omega}$ to Q' to calculate $\{y_t^j\}_{j\in\Omega}$. Then Q calculates and outputs $\{\nabla_a Q(s, a|\theta^Q)|_{s=s_t^j, a=\mu(s_t^j)}\}_{j\in\Omega}$ to μ. μ updates its parameters by (22). Then two soft updates are executed for Q' and μ' in (16) and (17), respectively.

The data flow of the critic's target Q-network Q' and online Q-network Q are as follows. Q' takes $\{(r_t^j, s_{t+1}^j, \mu'(s_{t+1}^j|\theta^{\mu'}))\}_{j\in\Omega}$ as the input and outputs

$\{y_t^j\}_{j\in\Omega}$ to Q. y_t^j is calculated by

$$y_t^j = r_t^j + \gamma Q'(s_{t+1}^j, \mu'(s_{t+1}^j|\theta^{\mu'})|\theta^{Q'}). \quad (20)$$

Q takes $\Big\{\{s_t^j, a_t^j\}_{j\in\Omega},\Big\}$ as the input and outputs $\{\nabla_a Q(s,a|\theta^Q)|_{s=s_t^j,a=\mu(s_t^j)}\}_{j\in\Omega}$ to μ for updating parameters in (22), where $\{s_t^j\}_{j\in\Omega}$ are sampled from R_b, and $\mu(s_t^j) = \arg\max_a Q(s_t^j, a)$.

The data flows of the actor's online policy network μ and target policy network μ' are as follows. After Q outputs $\{\nabla_a Q(s,a|\theta^Q)|_{s=s_t^j,a=\mu(s_t^j)}\}_{j\in\Omega}$ to μ, μ updates its parameters by (22). μ' takes $\{r_t^j, s_{t+1}^j\}_{j\in\Omega}$ as the input and outputs $\{r_t^j, s_{t+1}^j, \mu'(s_{t+1}^j|\theta^{\mu'})\}_{j\in\Omega}$ to Q' for calculating $\{y_t^j\}_{j\in\Omega}$ in (20), where $\{r_t^j, s_{t+1}^j\}_{j\in\Omega}$ are sampled from R_b.

The updates of parameters of four neural networks (Q, Q', μ, and μ') are as follows. The online Q-network Q updates its parameters by minimizing the L_2-norm loss function $\text{Loss}_t(\theta^Q)$ to make its Q-value fit y_t^j:

$$\nabla_{\theta^Q}\text{Loss}_t(\theta^Q) = \nabla_{\theta^Q}[\frac{1}{M}\sum_{j=1}^{M}(y_t^j - Q(s_t^j, a_t^j|\theta^Q))^2]. \quad (21)$$

The target Q-network Q' updates its parameters $\theta^{Q'}$ by (16). The online policy network μ updates its parameters following the chain rule with respect to θ^μ:

$$\begin{aligned}&\mathbb{E}_{s_t}[\nabla_{\theta^\mu} Q(s,a|\theta^Q)|_{s=s_t,a=\mu(s_t|\theta^\mu)}]\\ &\quad= \mathbb{E}_{s_t}[\nabla_a Q(s,a|\theta^Q)|_{s=s_t,a=\mu(s_t)}\nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s=s_t}]. \quad (22)\end{aligned}$$

The target policy network μ' updates its parameters $\theta^{\mu'}$ by (17).

In each time slot t, the current state s_t from the environment is delivered to μ', and μ' calculates the UAV's target policy $\mu'(s_t)|\theta^{\mu'}$. Finally, an exploration noise $\mathcal{N}$ is added to $\mu'(s_t|\theta^{\mu'})$ to get the UAV's action in (18).

2) In the test stage, we restore the neural network of the actor's target policy network μ' based on the stored parameters. This way, there is no need to store transitions to the experience replay buffer R_b. Given the current state s_t, we use μ' to obtain the UAV's optimal action $\mu'(s_t|\theta^{\mu'})$. Note that there is no noise added to $\mu'(s_t|\theta^{\mu'})$, since all neural networks have been trained and the UAV has got the optimal action through μ'. Finally, the UAV executes the action $\mu'(s_t|\theta^{\mu'})$.

E. Extension on Energy Consumption of 3D Flight

The UAV's energy is used in two parts, communication and 3D flight. The above proposed solutions in Alg. 3 do not consider the energy consumption of 3D flight. In this subsection, we discuss how to incorporate the energy consumption of 3D flight into Alg. 3. To encourage or discourage the UAV's 3D flight actions in different directions with different amount of energy consumption, we modify the reward function and the DDPG framework.

The UAV aims to maximize the total throughput per energy unit since the UAV's battery has limited capacity. For example, the UAV DJI Mavic Air [41] with full energy can only fly 21 minutes. Given that the UAV's energy consumption of 3D flight is much larger than that of communication, we only use the former part as the total energy consumption. Thus, the reward function (5) is modified as follows

$$\bar{r}(s_t, a_t) = \frac{1}{e(a_t)}\sum_{i\in\{0,1,2,3,4\}} bn_t^i c_t^i \log(1 + \frac{\rho_t^i h_t^i}{bc_t^i\sigma^2}), \quad (23)$$

where $e(a_t)$ is the energy consumption of taking action a_t in time slot t. Our energy consumption setups follow the UAV DJI Mavic Air [41]. The UAV has three vertical flight actions per time slot just as in (6). If the UAV keeps moving downward, horizontally, or upward until the energy for 3D flight is used up, the flight time is assumed to be 27, 21, and 17 minutes, respectively. If the duration of a time slot is set to 6 seconds, so the UAV can fly 270, 210, and 170 time slots, respectively. Therefore, the formulation of $e(a_t)$ is given by

$$e(a_t) = \begin{cases}\frac{1}{270}E_{\text{full}}, & \text{if moving downward 5 meters,}\\ \frac{1}{210}E_{\text{full}}, & \text{if moving horizontally,}\\ \frac{1}{170}E_{\text{full}}, & \text{if moving upward 5 meters,}\end{cases} \quad (24)$$

where E_{full} is the total energy if the UAV's battery is full.

Let $\delta(t)$ be a prediction error as follows

$$\delta(t) = \bar{r}(s_t, a_t) - Q(s_t, a_t), \quad (25)$$

where $\delta(t)$ evaluates the difference between the actual reward $\bar{r}(s_t, a_t)$ and the expected return $Q(s_t, a_t)$. To make the UAV learn from the prediction error $\delta(t)$, not the difference between the new Q-value and old Q-value in (14), the Q-value is updated by the following rule

$$\begin{aligned}&Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\delta(t) \Leftrightarrow\\ &Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(\bar{r}(s_t, a_t) - Q(s_t, a_t)), \quad (26)\end{aligned}$$

where α is a learning rate similar to (14).

We introduce α^+ and α^- to represent the learning rate when $\delta(t) \geq 0$ and $\delta(t) < 0$, respectively. Therefore, the UAV can choose to be active or inactive by properly setting the values of α^+ and α^-. The update of Q-value in Q-learning is modified as follows, inspired by [42]

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \begin{cases}\alpha^+\delta(t), & \text{if } \delta(t) \geq 0,\\ \alpha^-\delta(t), & \text{if } \delta(t) < 0.\end{cases} \quad (27)$$

We define the prediction error $\delta(t)$ as the difference between the actual reward and the output of the critic's online Q-network Q:

$$\delta(t) = \bar{r}(s_t, a_t) - Q(s_t, a_t|\theta^Q). \quad (28)$$

We use τ^+ and τ^- to denote the weights when $\delta(t) \geq 0$ and $\delta(t) < 0$, respectively. The update of the critic's target Q-network Q' is

$$\theta^{Q'} \leftarrow \begin{cases}\tau^+\theta^Q + (1-\tau^+)\theta^{Q'}, & \text{if } \delta(t) \geq 0,\\ \tau^-\theta^Q + (1-\tau^-)\theta^{Q'}, & \text{if } \delta(t) < 0.\end{cases} \quad (29)$$

The update of the actor's target policy network μ' is

$$\theta^{\mu'} \leftarrow \begin{cases}\tau^+\theta^\mu + (1-\tau^+)\theta^{\mu'}, & \text{if } \delta(t) \geq 0,\\ \tau^-\theta^\mu + (1-\tau^-)\theta^{\mu'}, & \text{if } \delta(t) < 0.\end{cases} \quad (30)$$

If $\tau^+ > \tau^-$, the UAV is active and prefers to move. If $\tau^+ < \tau^-$, the UAV is inactive and prefers to stay. If $\tau^+ = \tau^-$,

the UAV is neither active nor inactive. To approximate the Q-value, we introduce $\bar{y}_t^j$ similar to (20) and then make the critic's online Q-network Q to fit it. We optimize the loss function

$$\nabla_{\theta^Q}\text{Loss}_t(\theta^Q) = \nabla_{\theta^Q}[\frac{1}{M}\sum_{j=1}^{M}(\bar{y}_t^j - Q(s_t^j, a_t^j|\theta^Q))^2], \quad (31)$$

where $\bar{y}_t^j = \bar{r}_t^j$.

We modify the MDP, DDPG framework, and DDPG-based algorithms by considering the energy consumption of 3D flight:

- The MDP is modified as follows. The state space $\mathcal{S} = (L, x, z, n, H, E)$, where E is the energy in the UAV's battery. The energy changes as follows

$$E_{t+1} = \max\{E_t - e(a_t), 0\}. \quad (32)$$

  The other parts of MDP formulation and state transitions are the same as in Section III-C.
- There are three modifications in the DDPG framework: a) The critic's target Q-network Q' feeds $\bar{y}^j = \bar{r}^j$ to the critic's online Q-network Q instead of y^j in (20). b) The update of the critic's target Q-network Q' is (29) instead of (16). c) The update of the actor's target policy network μ' is (30) instead of (17).
- The DDPG-based algorithms are modified from Alg. 3. Initialize the energy state of the UAV as full in the start of each episode. In each time step of an episode, the energy state is updated by (32), and this episode terminates if the energy state $E_t \leq 0$. The reward function is replaced by (23).

V. Performance Evaluation

For a one-way-two-flow road intersection in Fig. 2, we present the optimality verification of deep reinforcement learning algorithms. Then, we study a more realistic road intersection as shown in Fig. 7, and present our simulation results.

Our simulations are executed on a server with Linux OS, 200 GB memory, two Intel(R) Xeon(R) Gold 5118 CPUs@2.30 GHz, a Tesla V100-PCIE GPU.

The implementation of Alg. 3 includes two parts: building the environment (including traffic and communication models) for our scenarios, and using the DDPG algorithm in TensorFlow [43].

A. Optimality Verification of Deep Reinforcement Learning

The parameter settings are summarized in Table II. In the simulations, there are three types of parameters: DDPG algorithm parameters, communication parameters, and UAV/vehicle parameters.

First, we describe the DDPG algorithm parameters. The number of episodes is 256, and the number of time slots in an episode is 256, so the number of total time slots is 65,536. The experience replay buffer capacity is 10,000, and the learning rate of target networks τ is 0.001. The mini-batch size M is 512. The training data set is full in the $10,000^{th}$ time slot, and is updated in each of the following 256 × 256 - 10,000 = 55,536 time slots. The test data set is real-time among all the 256 × 256 = 65,536 time slots.

TABLE II
Values of parameters in simulation settings

α_1	α_2	β_1	β_2	σ^2	$\widehat{d}$
9.6	0.28	3	0.01	-130 dBm/Hz	3
P	C	N	γ	$z_{\min}$	$z_{\max}$
1 ∼ 6	10	10	0.9	10	200
M	λ	g_i^s	g_i^l	g_i^r	b
512	0.1 ∼ 0.7	0.4	0.3	0.3	100 KHz
τ	$\rho_{\max}$	$c_{\max}$			
0.001	3 W	5			

Secondly, we describe communication parameters. α_1 and α_2 are set to 9.6 and 0.28, which are common values in urban areas [44]. β_1 is 3, and β_2 is 0.01, which are widely used in path loss modeling. The duration of a time slot is set to 6 seconds, and the number of occupied red or green traffic light N is 10, i.e., 60 seconds constitute a red/green duration, which is commonly seen in cities and can ensure that the vehicles in blocks can get the next block in a time slot. The white power spectral density σ^2 is set to -130 dBm/Hz. The total UAV transmission power P is set to 6 W in consideration of the limited communication ability. The total number of channels C is 10. The bandwidth of each channel b is 100 KHz. Therefore, the total bandwidth of all channels is 1 MHz. The maximum power allocated to a vehicle $\rho_{\max}$ is 3 W, and the maximum number of channels allocated to a vehicle $c_{\max}$ is 5. We assume that the power control for each vehicle has 4 discrete values (0, 1, 2, 3).

Thirdly, we describe UAV/vehicle parameters. λ is set to 0.1 ∼ 0.7. The length of a road block $\widehat{d}$ is set to 3 meters. The blocks' distance is easily calculated as follows: $D(1,0) = \widehat{d}$, and $D(1,3) = 2\widehat{d}$, where $D(i,j)$ is the Euclidean distance from block i to block j. We assume the arrival of vehicles in block 1 and 2 follows a binomial distribution with the same parameter λ in the range $0.1 \sim 0.7$. The discount factor γ is 0.9.

The assumptions of the simplified scenario in Fig. 2 are as follows. To keep the state space small for verification purpose, we assume the channel states of all communication links are LoS, and the UAV's height is fixed as 150 meters, so that the UAV can only adjusts its horizontal flight control and transmission control. The traffic light state is assumed to have two values (red or green).

The configure of neural networks in proposed solutions is based on the configure of the DDPG action space. A neural network consists of an input layer, fully-connected layers, and an output layer. The number of fully-connected layers in actor is set to 4.

Theoretically, it is well-known that deep reinforcement learning algorithms (including DDPG algorithms) solve MDP problems and achieve the optimal results with much less memory and computational resources. We provide the optimality verification of DDPG-based algorithms in Alg. 3 in a one-way-two-flow road intersection in Fig. 2. The reasons are as follows: (i) the MDP problem in such a simplified scenario is explicitly defined and the theoretically optimal

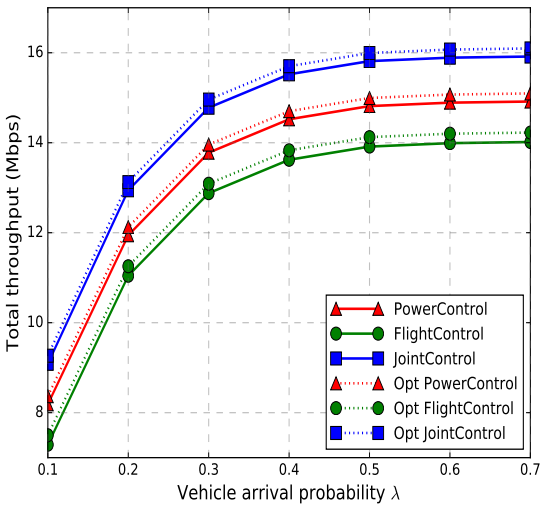

Fig. 6. Total throughput vs. vehicle arrival probability λ in optimality verification.

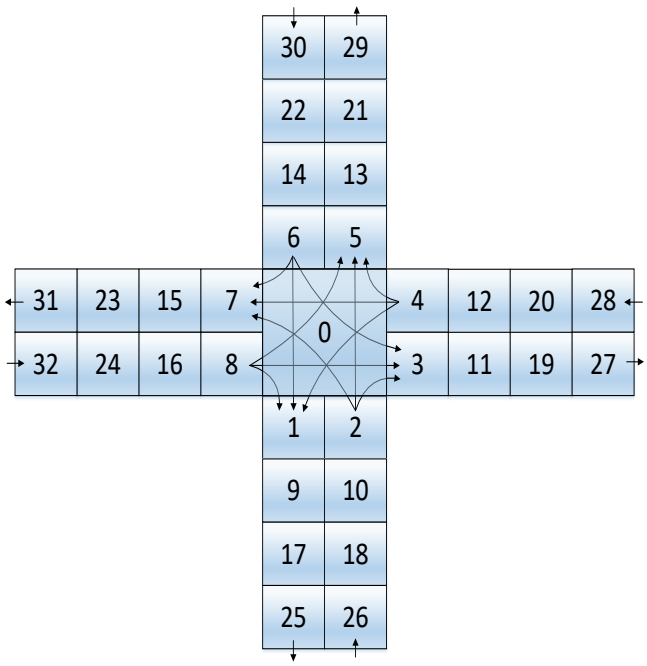

Fig. 7. Realistic road intersection model.

policy can be obtained using the Python MDP Toolbox [45]; and (ii) this optimality verification process also serves a good code debugging process before we apply the DDPG algorithm in TensorFlow [43] to the more realistic road intersection scenario in Fig. 7.

The result of DDPG-based algorithms matches that of the policy iteration algorithm using Python MDP Toolbox [45] (serving as the optimal policy). The total throughput obtained by the policy iteration algorithm and DDPG-based algorithms are shown as dashed lines and solid lines in Fig. 6. Therefore, DDPG-based algorithms achieve near optimal policies. We see that, the total throughput in JointControl is the largest, which is much higher than PowerControl and FlightControl. This is in consistent with our believes that the JointControl of power and flight allocation will be better than the control of either of both. The performance of PowerControl is better than FlightControl. The throughput increases with the increasing of vehicle arrival probability λ in all algorithms, and it saturates when $\lambda \geq 0.6$ due to traffic congestion.

B. More Realistic Traffic Model

We consider a more realistic road intersection model in Fig. 7. There are totally 33 blocks with four entrances (block 26, 28, 30, and 32), and four exits (block 25, 27, 29, and 31). Vehicles in block $i \in \{2, 4, 6, 8\}$ go straight, turn left, turn right with the probabilities g_i^s, g_i^l, and g_i^r, such that $g_i^s + g_i^l + g_i^r = 1$. We assume vehicles can turn right when the traffic light is green.

Now, we describe the settings different from the last subsection. The discount factor γ is $0.4 \sim 0.9$. The total UAV transmission power P is set to $1 \sim 6$ W. The total number of channels C is $100 \sim 200$, which is much larger than that in subsection V-A since there are more vehicles in the realistic model. The bandwidth of each channel b is 5 KHz, therefore, the total bandwidth of all channels is $0.5 \sim 1$ MHz. The maximum power allocated to a vehicle $\rho_{\max}$ is 0.9 W, and the maximum number of channels allocated to a vehicle $c_{\max}$ is 50. The minimum and maximum height of the UAV is 10 meters and 200 meters. The probability of a vehicle going straight, turning left, and turning right (g_i^s, g_i^l, and g_i^r) is set to 0.4, 0.3, and 0.3, respectively, and each of them is assumed to be the same in block 2, 4, 6, and 8. We assume the arrival of vehicles in block 26, 28, 30, and 32 follows a binomial distribution with the same parameter λ in the range $0.1 \sim 0.7$.

The UAV's horizontal and vertical flight actions are as follows. We assume that the UAV's block is $0 \sim 8$ since the number of vehicles in the intersection block 0 is generally the largest and the UAV will not move to the block far from the intersection block. Moreover, within a time slot we assume that the UAV can stay or only move to its adjacent blocks. The UAV's vertical flight action is set by (6). In PowerControl, the UAV stays at block 0 with the height of 150 meters.

C. Baseline Schemes

We compare with two baseline schemes. Generally, the equal transmission power and channels allocation is common in communication systems for fairness. Therefore, they are used in baseline schemes.

The first baseline scheme is Cycle, i.e., the UAV cycles anticlockwise at a fixed height (e.g., 150 meters), and the UAV allocates the transmission power and channels equally to each vehicle in each time slot. The UAV moves along the fixed trajectory periodically, without considering the vehicle flows.

The second baseline scheme is Greedy, i.e., at a fixed height (e.g., 150 meters), the UAV greedily moves to the block with the largest number of vehicles. If a nonadjacent block has the largest number of vehicles, the UAV has to move to block 0 and then move to that block. The UAV also allocates the transmission power and the channels equally to each vehicle

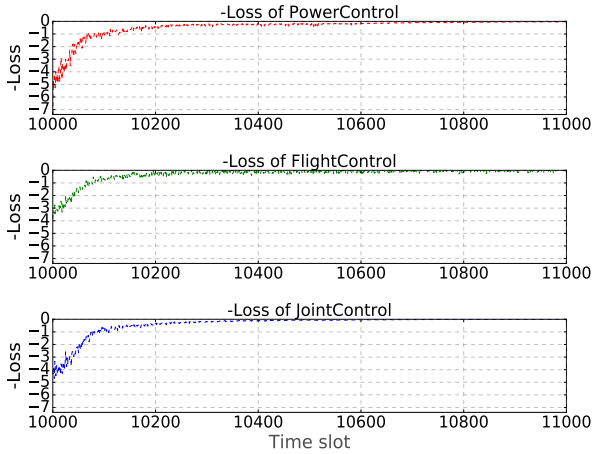

Fig. 8. Convergence of loss functions in training stage.

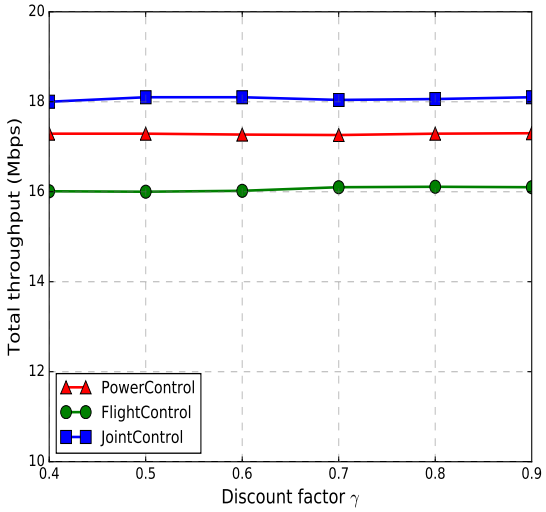

Fig. 9. Throughput vs. discount factor γ.

in each time slot. The UAV tries to serve the block with the largest number of vehicles by moving nearer to them.

D. Simulation Results

The training time is about 4 hours, and the test time is almost real-time, since it only uses the well trained target policy network. Next, we first show the convergence of loss functions, and then show total throughput vs. discount factor, total transmission power, total number of channels and vehicle arrival probability, and finally present the total throughput and the UAV's flight time vs. energy percent for 3D flight.

The convergence of loss functions in training stage for PowerControl, FlightControl, and JointControl indicates that the neural network is well-trained. It is shown in Fig. 8 when $P = 6$, $C = 200$, $\lambda = 0.5$ and $\gamma = 0.9$ during time slots 10,000 $\sim$ 11,000. The first 10,000 time slots are not shown since during the 0 $\sim$ 10,000, the experience replay buffer has not achieved its capacity. We see that, the loss functions in three algorithms converge after time slot 11,000. The other metrics in the paper are measured in test stage by default.

Total throughput vs. discount factor γ is drawn in Fig. 9 when $P = 6$, $C = 200$, and $\lambda = 0.5$. We can see that, when γ changes, the throughput of three algorithms is steady; and JointControl achieves higher total throughput, comparing with PowerControl and FlightControl, respectively. PowerControl achieves higher throughput than FlightControl since PowerControl allocates power and channel to strongest channels while FlightControl only adjusts the UAV's 3D position to enhance the strongest channel and the equal power and channel allocation is far from the best strategy in OFDMA.

Total throughput vs. total transmission power ($P = 1 \sim 6$) and total number of channels ($C = 100 \sim 200$) are shown by Fig. 10 and Fig. 11, where we set $\lambda = 0.5$ and $\gamma = 0.9$. We see that JointControl achieves the best performance for different transmission power and channel budgets, respectively. Moreover, the total throughput of all algorithms increases when the total transmission power or total number of channels increases. PowerControl and FlightControl only adjust the transmission power or 3D flight, while JointControl jointly

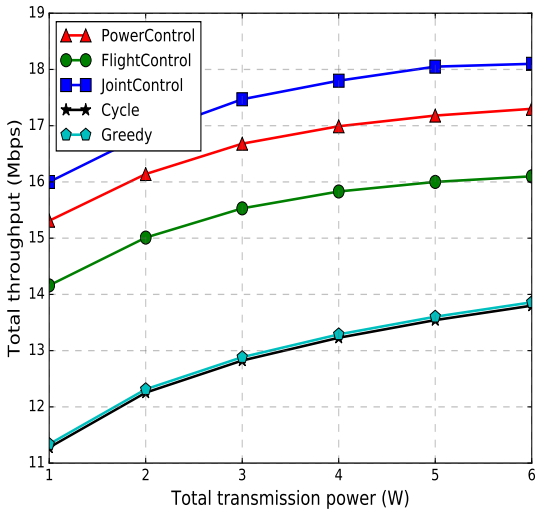

Fig. 10. Total throughput vs. total transmission power (C = 200).

adjusts both of them, so its performance is the best. The total throughput of DDPG-based algorithms is improved greatly than that of Cycle and Greedy. The performance of Greedy is a little better than Cycle, since Greedy tries to get nearer to the block with the largest number of vehicles.

Total throughput vs. vehicle arrival probability λ is shown in Fig. 12. Note that the road intersection has a capacity of 2 units, i.e., it can serve at most two traffic flows at the same time, therefore, it cannot serve traffic flows where λ is very high, e.g., $\lambda = 0.8$ and $\lambda = 0.9$. We see that, when λ increases, i.e., more vehicles arrive at the intersection, the total throughput increases. However, when λ gets higher, e.g., $\lambda = 0.6$, the total throughput saturates due to traffic congestion.

Next, we test the metrics considering of the energy consumption of 3D flight. The total throughput vs. energy percent

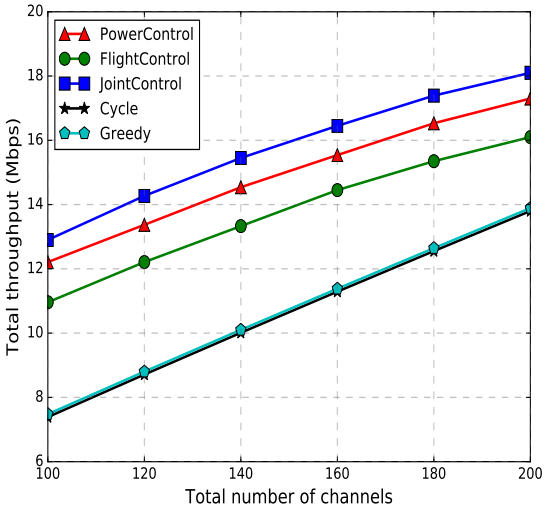

Fig. 11. Total throughput vs. total number of channels (P = 6).

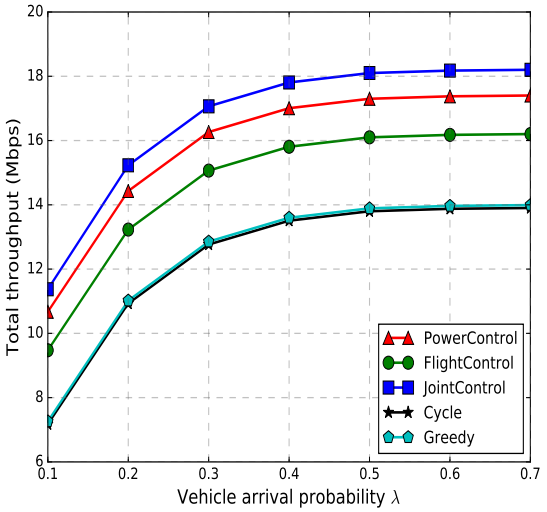

Fig. 12. Total throughput vs. vehicle arrival probability λ.

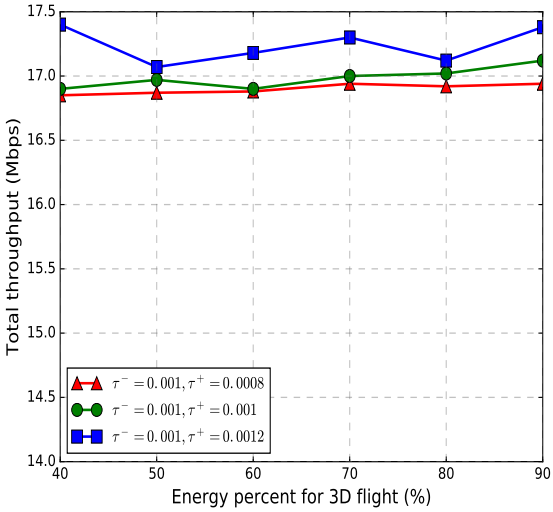

Fig. 13. Total throughput vs. energy percent for 3D flight in JointControl (P = 6, C = 200).

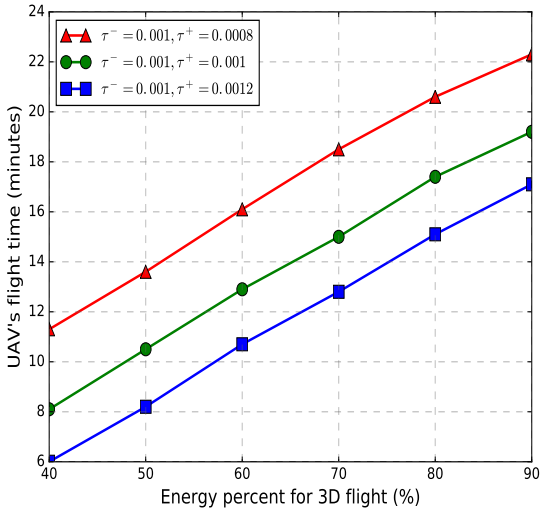

Fig. 14. UAV's flight time vs. energy percent for 3D flight in JointControl (P = 6, C = 200).

for 3D flight in JointControl is shown in Fig. 13. When τ^+ increases, the total throughput almost increases. We get that if the UAV is more active in 3D flight, it will help to improve the total throughput. However, the improvement of the total throughput is not very clear since the UAV has to consider the energy consumption in the new reward function (23). In addition, when τ^+ is higher, the total throughput has more variance since the UAV prefers to get higher reward through more ventures.

The UAV's flight time vs. energy percent for 3D flight in JointControl is shown in Fig. 14. When $\tau^- = 0.001$ and $\tau^+ = 0.0008$, the UAV's flight time is the longest since the UAV is inactive. When $\tau^- = 0.001$ and $\tau^+ = 0.0012$, the UAV's flight time is the shortest, since the UAV is active and prefers to flight. When $\tau^- = \tau^+ = 0.001$, the UAV's flight time is between the other two cases. If the energy percent for 3D flight increases, the UAV's flight time increases linearly in the three cases.

VI. Conclusions

We studied a UAV-assisted vehicular network where the UAV acted as a relay to maximize the total throughput between the UAV and vehicles. We focused on the downlink communication where the UAV could adjust its transmission control (power and channel) under 3D flight. We formulated our problem as a MDP problem, explored the state transitions of UAV and vehicles under different actions, and then proposed three deep reinforcement learning schemes based on the DDPG algorithms, and finally extended them to account for the energy consumption of the UAV's 3D flight by modifying

the reward function and the DDPG framework. In a simplified scenario with small state space and action space, we verified the optimality of DDPG-based algorithms. Through simulation results, we demonstrated the superior performance of the algorithms under a more realistic traffic scenario compared with two baseline schemes.

In the future, we will consider the scenario where multiple UAVs constitute a relay network to assist vehicular networks and study the coverage overlap/probability, relay selection, energy harvesting communications, and UAV cooperative communication protocols. We pre-trained the proposed solutions using servers, and we hope the UAV trains the neural netwroks in the future if light and low energy consumption GPUs are applied at the edge.

REFERENCES

[1] M. Chaqfeh, H. El-Sayed, and A. Lakas, "Efficient data dissemination for urban vehicular environments," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, no. 99, pp. 1–11, 2018.

[2] M. Zhu, X.-Y. Liu, F. Tang, M. Qiu, R. Shen, W. Shu, and M.-Y. Wu, "Public vehicles for future urban transportation," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, vol. 17, no. 12, pp. 3344–3353, 2016.

[3] M. Zhu, X.-Y. Liu, and X. Wang, "Joint transportation and charging scheduling in public vehicle systems - a game theoretic approach," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, vol. 19, no. 8, pp. 2407–2419, 2018.

[4] M. Zhu, X.-Y. Liu, and X. Wang, "An online ride-sharing path-planning strategy for public vehicle systems," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, vol. 20, no. 2, pp. 616–627, 2019.

[5] K. Li, C. Yuen, S. S. Kanhere, K. Hu, W. Zhang, F. Jiang, and X. Liu, "An experimental study for tracking crowd in smart cities," *IEEE Systems Journal*, 2018.

[6] F. Cunha, L. Villas, A. Boukerche, G. Maia, A. Viana, R. A. Mini, and A. A. Loureiro, "Data communication in VANETs: protocols, applications and challenges," *Elsevier Ad Hoc Networks*, vol. 44, pp. 90–103, 2016.

[7] H. Sedjelmaci, S. M. Senouci, and N. Ansari, "Intrusion detection and ejection framework against lethal attacks in UAV-aided networks: a bayesian game-theoretic methodology," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, vol. 18, no. 5, pp. 1143–1153, 2017.

[8] "Paving the path to 5G: optimizing commercial LTE networks for drone communication (2018)." https://www.qualcomm.cn/videos/paving-path-5g-optimizing-commercial-lte-networks-drone-communication.

[9] "Huawei signs MoU with China Mobile Sichuan and Fonair aviation to build cellular test networks for logistics drones (2018)." https://www.huawei.com/en/press-events/news/2018/3/MoU-ChinaMobile-FonairAviation-Logistics.

[10] M. Alzenad, A. El-Keyi, F. Lagum, and H. Yanikomeroglu, "3-D placement of an unmanned aerial vehicle base station (UAV-BS) for energy-efficient maximal coverage," *IEEE Wireless Communications Letters (WCL)*, vol. 6, no. 4, pp. 434–437, 2017.

[11] M. Giordani, M. Mezzavilla, S. Rangan, and M. Zorzi, "An efficient uplink multi-connectivity scheme for 5G mmWave control plane applications," *IEEE Transactions on Wireless Communications (TWC)*, 2018.

[12] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Transactions on Cognitive Communications and Networking (TCCN)*, vol. 4, no. 2, pp. 257–265, 2018.

[13] Q. Yang and S.-J. Yoo, "Optimal UAV path planning: sensing data acquisition over IoT sensor networks using multi-objective bio-inspired algorithms," *IEEE Access*, vol. 6, pp. 13671–13684, 2018.

[14] M. Garraffa, M. Bekhti, L. Létocart, N. Achir, and K. Boussetta, "Drones path planning for WSN data gathering: a column generation heuristic approach," in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2018.

[15] C. H. Liu, Z. Chen, J. Tang, J. Xu, and C. Piao, "Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 36, no. 9, pp. 2059–2070, 2018.

[16] H. Wang, G. Ding, F. Gao, J. Chen, J. Wang, and L. Wang, "Power control in UAV-supported ultra dense networks: communications, caching, and energy transfer," *IEEE Communications Magazine*, vol. 56, no. 6, pp. 28–34, 2018.

[17] S. Yan, M. Peng, and X. Cao, "A game theory approach for joint access selection and resource allocation in UAV assisted IoT communication networks," *IEEE Internet of Things Journal (IOTJ)*, 2018.

[18] Y. Wu, J. Xu, L. Qiu, and R. Zhang, "Capacity of UAV-enabled multicast channel: joint trajectory design and power allocation," in *IEEE International Conference on Communications (ICC)*, pp. 1–7, 2018.

[19] Y. Zeng, X. Xu, and R. Zhang, "Trajectory design for completion time minimization in UAV-enabled multicasting," *IEEE Transactions on Wireless Communications (TWC)*, vol. 17, no. 4, pp. 2233–2246, 2018.

[20] S. Zhang, Y. Zeng, and R. Zhang, "Cellular-enabled UAV communication: trajectory optimization under connectivity constraint," in *IEEE International Conference on Communications (ICC)*, pp. 1–6, 2018.

[21] R. Fan, J. Cui, S. Jin, K. Yang, and J. An, "Optimal node placement and resource allocation for UAV relaying network," *IEEE Communications Letters*, vol. 22, no. 4, pp. 808–811, 2018.

[22] U. Challita, W. Saad, and C. Bettstetter, "Deep reinforcement learning for interference-aware path planning of cellular-connected UAVs," in *IEEE International Conference on Communications (ICC)*, 2018.

[23] X.-Y. Liu, Z. Ding, S. Borst, and A. Walid, "Deep reinforcement learning for intelligent transportation systems," in *NeurIPS Workshop on Machine Learning for Intelligent Transportation Systems*, 2018.

[24] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Communications Letters (WCL)*, vol. 3, no. 6, pp. 569–572, 2014.

[25] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications (TWC)*, vol. 15, no. 6, pp. 3949–3963, 2016.

[26] D. Oehmann, A. Awada, I. Viering, M. Simsek, and G. P. Fettweis, "SINR model with best server association for high availability studies of wireless networks," *IEEE Wireless Communications Letters (WCL)*, vol. 5, no. 1, pp. 60–63, 2015.

[27] N. Gupta and V. A. Bohara, "An adaptive subcarrier sharing scheme for ofdm-based cooperative cognitive radios," *IEEE Transactions on Cognitive Communications and Networking (TCCN)*, vol. 2, no. 4, pp. 370–380, 2016.

[28] P. Ramezani and A. Jamalipour, "Throughput maximization in dual-hop wireless powered communication networks," *IEEE Transactions on Vehicular Technology (TVT)*, vol. 66, no. 10, pp. 9304–9312, 2017.

[29] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[30] Q. Wu and R. Zhang, "Common throughput maximization in uav-enabled ofdma systems with delay consideration," *IEEE Transactions on Communications (TOC)*, vol. 66, no. 12, pp. 6614–6627, 2018.

[31] S. Zhang, Y. Zeng, and R. Zhang, "Cellular-enabled uav communication: Trajectory optimization under connectivity constraint," in *IEEE International Conference on Communications (ICC)*, pp. 1–6, 2018.

[32] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[33] C. J. Watkins and P. Dayan, "Q-learning," *Springer Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[34] C. Wirth and G. Neumann, "Model-free preference-based reinforcement learning," in *AAAI Conference on Artificial Intelligence*, 2016.

[35] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *AAAI Conference on Artificial Intelligence*, 2016.

[36] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2016.

[37] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[38] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *https://arxiv.org/pdf/1312.5602*, 2013.

[39] A. Daniely, "SGD learns the conjugate kernel class of the network," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 2422–2430, 2017.

[40] Z. Wang, V. Aggarwal, and X. Wang, "Joint energy-bandwidth allocation in multiple broadcast channels with energy harvesting," *IEEE Transactions on Communications (TOC)*, vol. 63, no. 10, pp. 3842–3855, 2015.
[41] "Homepage of DJI Mavic Air (2019)." https://www.dji.com/cn/mavic-air?site=brandsite&from=nav.
[42] G. Lefebvre, M. Lebreton, F. Meyniel, S. Bourgeois-Gironde, and S. Palminteri, "Behavioural and neural characterization of optimistic reinforcement learning," *Nature Human Behaviour*, vol. 1, no. 4, p. 0067, 2017.
[43] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: a system for large-scale machine learning," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 265–283, 2016.
[44] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Drone small cells in the clouds: design, deployment and performance analysis," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2015.
[45] "Python Markov decision process (MDP) Toolbox (2019)." https://pymdptoolbox.readthedocs.io/en/latest/api/mdptoolbox.html.

Ming Zhu received the Ph.D. degree in Computer Science and Engineering in Shanghai Jiao Tong University, Shanghai, China. He is now a Post-Doctoral Researcher and an Assistant Professor in Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China.

His research interests are in the area of big data, intelligent transportation systems, smart cities, and artificial intelligence.

Xiao-Yang Liu received the B.Eng. degree in Computer Science from Huazhong University of Science and Technology, and the PhD degree in the Department of Computer Science and Engineer, Shanghai Jiao Tong University, China. He is currently a PhD in the Department of Electrical Engineering, Columbia University.

His research interests include tensor theory, deep learning, non-convex optimization, big data analysis and IoT applications.

Xiaodong Wang (S'98-M'98-SM'04-F'08) received the Ph.D. degree in electrical engineering from Princeton University. He is currently a Professor of electrical engineering with Columbia University, New York NY, USA. His research interests fall in the general areas of computing, signal processing, and communications. He has authored extensively in these areas. He has authored the book entitled Wireless Communication Systems: Advanced Techniques for Signal Reception, (Prentice Hall, 2003). His current research interests include wireless communications, statistical signal processing, and genomic signal processing. He has served as an Associate Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON SIGNAL PROCESSING, and IEEE TRANSACTIONS ON INFORMATION THEORY. He is an ISI Highly Cited Author. He received the 1999 NSF CAREER Award, the 2001 IEEE Communications Society and Information Theory Society Joint Paper Award, and the 2011 IEEE Communication Society Award for Outstanding Paper on New Communication Topics.