Maintaining Links in the Highly Dynamic FANET Using Deep Reinforcement Learning

Xiulin Qiu, *Student Member, IEEE*, Yuwang Yang, Lei Xu, *Member, IEEE*, Jun Yin, and Zhenqiang Liao

***Abstract*—Routing protocols do not respond quickly to environmental changes due to the high mobility of nodes in the Flying Ad Hoc Network (FANET), to obtain reliable transmission links. This paper proposes an adaptive link maintenance method based on deep reinforcement learning (DRL-MLsA), which can dynamically adjust the time interval of broadcasting Hello packets. This method can cope with the highly dynamic network environment, and adapt to both active routing and table-driven routing protocols. The method considers the channel model of the signal and investigates the impact of UAV communication range on link maintenance. We can get an agent by perceiving the degree of changes in the number of neighbors in a dynamic environment. The optimal broadcast cycle was obtained to maximize the energy of the node to send and receive task data. We substituted the single-output network model with a competitive network to overcome the reward overestimation problem, which also improves the convergence speed of the algorithm. Simulation results showed that DRL-MLsA can reduce the communication overhead for link maintenance, while at the same time increase the throughput of the network and decrease the packet loss of transmission.**

***Index Terms*—FANET, highly dynamic environment, link maintenance, deep reinforcement learning.**

I. Introduction

The development of unmanned aerial vehicle (UAV) technology has led to the emergence of UAV-assisted services, such as battlefield environment monitoring, rapid reconnaissance response, battlefield situation assessment, commercial Internet, scientific investigation, and health surveillance [1], [2], [3], [4], [5]. The Flying Ad Hoc Network (FANET) is characterized by self-organization, information sharing, and strong scalability, which can provide a fundamental guarantee for UAV-assisted tasks [6], [7]. Considering the application environment of high-speed movement and dynamic changes in topology, the connectivity of flying nodes is one focus of FANET research. Traditional table-driven routing protocols are restricted, and pre-established routes may fail quickly. Therefore, nodes need to maintain links in a changing environment.

UAVs usually perform link maintenance in FANET by broadcasting Hello packets [8]. In a highly dynamic environment, as shown in Fig. 1, link maintenance based on a fixed cycle can no longer meet the requirements of the flight mission. 1) UAVs can travel to any location at any speed during the flight mission, which will affect the network topology, thus adversely affecting the communications network throughput, leading to data distribution delays. If more reliable transmission links are obtained by simply reducing the transmission cycle of link maintenance packets, and repeatedly trying to broadcast Hello packets to discover neighbor nodes, this will increase the burden of link maintenance and consume a great deal of node energy [9]. 2) The mobility of UAVs will cause the optimal link for data transmission to be missed. If signaling packets are exchanged at a fixed rate, when a node sends a data packet to its neighbor, the neighbor may no longer be in the same position or may be directly missed, and the data packet will be directly lost [10]. When the mobility of UAVs changes due to switching in the types of task (e.g., the search task of UAV is switched to a tracking task), fixed-cycle link maintenance no longer adapts to the existing network environment. 3) How the UAV balances the energy distribution between the data transmission and the control transmission has become a key issue. New link maintenance methods should be based on the perception of link performance and should adaptively adjust the cycle of neighbor node detection rather than adopt a fixed and single adjustment strategy. The problems outlined above have been studied extensively, and are introduced in detail in Section II. However, most of the solutions cannot be deployed quickly and efficiently without reducing network throughput.

We applied the concept of learning to the link maintenance strategy of FANET. The results of machine learning algorithms can provide perceptual information for unknown topological changes. However, in practical applications, a series of labeled datasets that conform to the actual situation must be provided, which is difficult for network researchers; reinforcement learning [11] effectively avoids the above problem as it generates training data through real-time interaction with the environment, and learns strategies for adapting to the changing environment by formulating reward functions [12], [13], [14]. In this paper, reinforcement learning was used to adaptively adjust the time

Manuscript received 28 November 2020; revised 29 July 2021, 23 November 2021, 28 April 2022, and 3 July 2022; accepted 19 October 2022. Date of publication 28 October 2022; date of current version 14 March 2023. This work was supported in part by NDF under Grants JCKY201760**003 and JCKY201860**001, in part by JSKS&T under Grant BE2018393, and in part by SZKITI under Grant SYG201826. The review of this article was coordinated by Prof. Yaser P. Fallah. *(Corresponding author: Yuwang Yang.)*

Xiulin Qiu, Yuwang Yang, and Lei Xu are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: qiuxiulin@njust.edu.cn; yuwangyang@njust.edu.cn; xulei_marcus@126.com).

Jun Yin is with the Jiangsu Key Laboratory for Broadband Wireless Communication and Internet of Things, Nanjing University of Post and Telecommunications, Nanjing 210046, China (e-mail: junyin@njupt.edu.cn).

Zhenqiang Liao is with the School of Electrical and Mechanical Engineering, Suzhou Global Institute of Software Technology, Suzhou 215163, China (e-mail: zqliao1013@126.com).

Digital Object Identifier 10.1109/TVT.2022.3217888

interval of each node in FANET to broadcast Hello packets, so that the optimal link maintenance cycle can be determined automatically. Specifically, the goal of learning was not to predict the network environment at the next moment and find the corresponding optimal time interval for sending a Hello packet, but to learn the mapping between the optimal time interval at the next moment and the observed link state at this moment. However, when there are too many ambient states to be exhaustive, it is necessary to use function approximation for fitting. Deep learning has been implemented in a wide range of fields over the past decade. DeepMind [15] proposed deep reinforcement learning (DRL) in 2015 by combining the strong perception ability of deep learning with the decision-making ability of reinforcement learning, which effectively solved the problem of strategy learning under continuous or infinite states [16], [17], [18], [19], [20].

In this paper, a new method of link maintenance was proposed, and the main contributions of this article are summarized as follow:

1) A method of adaptively adjusting the link maintenance cycle in the routing protocol was proposed, which uses DRL for modeling and adapts to all active routing protocols;
2) To solve problems such as slow convergence of the training algorithm and large computational consumption, the network model and value function have been improved to accelerate training convergence and increase stability;
3) The agent obtained through UAV loading learning does not require additional computation in a changing network environment. The enhanced routing protocol makes it possible for nodes to get more transmission links, thereby increasing network throughput and reducing packet loss.

The rest of this paper proceeds as follows. Section II introduces related research regarding link maintenance. Various parts of the system are modeled in Section III. Section IV proposes an improved training algorithm. A test is discussed in Section V to confirm the effectiveness of the model, and Section VI presents the discussion and conclusion.

II. Related Work

A number of groups have proposed solutions for determining the interval of neighbor detection in the FANET routing protocol. The ideas behind these solutions are discussed in this section.

Hernandez et al. [21] improved the traditional link maintenance method of sending messages based on a fixed cycle, and designed a utility function of the link change rate to reflect the network status, which was used to adjust dynamically the rate at which each node sends control messages. Although this method takes the impact of topological changes into consideration, the value of the link change rate must be calculated in real-time, and this value belongs to the network information of the global state, which increases the computational burden on nodes.

Han et al. [22] developed an adaptive Hello message passing solution to discover neighbors that use the average interval of events, i.e., the average time interval between two consecutive events (i.e., sending or receiving a data packet) on a node, to estimate how active the node is in sending or forwarding. If a node does not participate in any communication within a given time, there is no need to maintain the link-state, and the Hello packets broadcast during this period are unnecessary. If a constant Hello interval is used, with the increasing interval of events, the probability of attempting to transmit data packets through the disconnected link will be reduced. However, if two UAVs remain out of communication for a long time and encounter each other later, they will not consider each other as neighbors.

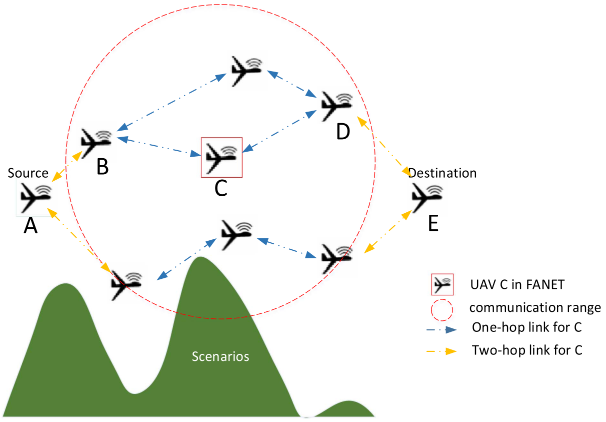


Fig. 1. FANET scenario in a highly dynamic environment.

Park [23] studied the influence of node velocity and transmission range on the Hello interval in MANET from the perspective of network throughput. Simulation of the MANET of the Ad hoc On-Demand Distance Vector (AODV) routing protocol showed that the Hello interval of the maximum network throughput can be determined as a function of node velocity and transmission range. This approach only considers the impacts of two factors on the link maintenance cycle, and is unsuitable for FANET.

Based on the available task-related information, such as the volume of allowable airspace, the number of UAVs, and the transmission range and speed of UAVs, Mahmud [24] proposed a novel adaptive Hello interval solution, Energy Saving Hello (EE-Hello), to avoid unnecessary energy consumption. This solution saved 25% of communication maintenance energy without reducing the overall network throughput. However, it is only suitable for tasks assisted by micro-UAVs, and for high-speed mobile large-scale UAV scenarios, the factors it considers are not the main factors.

Wei [25] learned the optimal TCP congestion control strategy online based on the reinforcement learning framework, and identified network congestion by perceiving the time interval between two ACK packets. Then, whether to increase or decrease the time interval between two packets was determined after the Q learning algorithm was solved to control network congestion.

Based on the above related work, the rate of Hello packet broadcast in FANET was adjusted in real time by the DRL solution in this study to cope with the highly dynamic network environment.

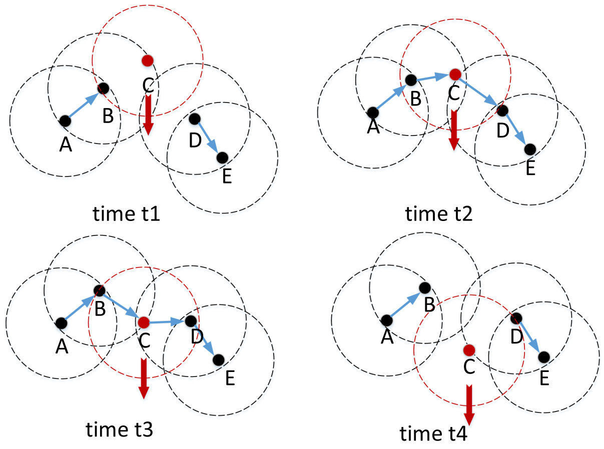


Fig. 2. The impact of UAV mobility on link maintenance.

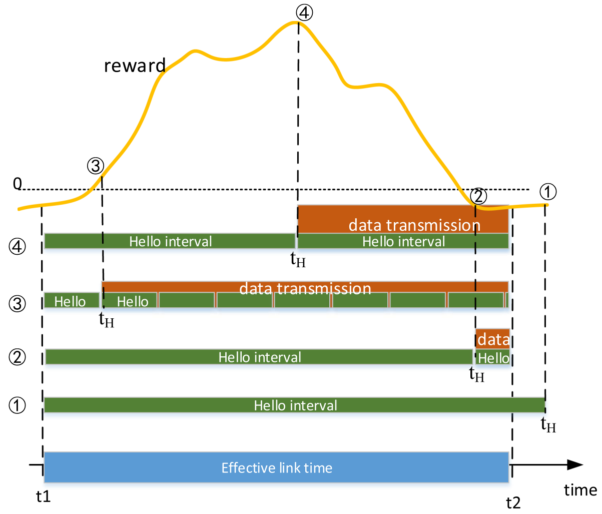


Fig. 3. Impact of adjusting the Hello packet period on FANET link resources.

III. System Model

This section will introduce the establishment process of the system model in detail, including the description of the problem, the definition of the state, action, and value function. Finally, we propose an overall framework for link maintenance based on deep reinforcement learning.

A. Research Motivation

To describe the method proposed in this paper, the highly dynamic FANET environment was analyzed. As shown in Fig. 2, in a highly dynamic mission scenario for UAVs, each UAV can move in any direction, at any speed, at any time. Suppose there are five UAVs (designated as A, B, C, D, and E) for information interaction, the dotted circle is the communication range of the UAVs. C (red solid circle) moves downward at a certain relative speed V, while the other four UAVs remain relatively stationary. Suppose several data packets need to be sent from A to E, at time t. Due to the relative movement of C, data cannot be transmitted at time t1, and network connection can only be established between time t2 and time t3. As the UAV continues to move, the communication link cannot be established after time t4.

Many routing protocols (e.g., AODV, OLSR) use a fixed period to broadcast Hello messages to detect neighboring nodes. After a node receives a Hello message from another node, it starts a timer; the link is considered valid for a certain period of time. If no Hello message is received within this period, the link is considered broken. The frequency of Hello message transmission depends on the mobility of the node: if the node moves quickly and Hello messages are rarely transmitted, neighboring nodes may be within communication range but will not be detected.

As shown in Fig. 3, with regard to the task of link maintenance, with the rapid movement of UAV, if the cycle of sending Hello packets is $T_H > (t_2 - t_1)$ (situation ① in Fig. 3), the demand for data packets to be sent from A to E cannot be satisfied. At this time, cycle T_H must be reduced, which means that it is necessary to broadcast Hello packets more frequently to obtain more neighbor nodes to ensure sufficient links for data transmission. However, due to the highly dynamic characteristics, faster broadcast of link maintenance information is not feasible as a large number of Hello packets will waste communication resources (situation ③ in Fig. 3).

Based on DRL, this study proposes a link maintenance strategy that adaptively adjusts the cycle of Hello packets broadcast, and the agent obtained by learning can solve the above problems. The agent repeatedly interacts with the highly dynamic network environment. In each time slot, t, the agent first perceives the current environment state, s_t ($s \in S$), and selects an action, a_t ($a \in A$), from a fixed set of actions. The environment changes with the action of the agent to generate state, s_{t+1}. At this time, the agent will acquire the reward value, r_t, from the environment to determine whether the action selected by the agent is good. After repeated learning in this way, as shown in the ④ situation in Fig. 3, the agent may obtain a mapping, π (i.e., $\pi : S \rightarrow A$), from the possible state, S, to the optimal action, A.

B. Channel Model

In this paper, we focus on networking among UAVs, so the channel is considered only for the UAV-to-UAV case. In different application scenarios, UAV channels exhibit varying properties. Currently, UAV links tend to be in the L-band (0.9–1.2 GHz) and C-band (5.03–5.091 GHz) [31], whereas data links usually use unlicensed Wi-Fi bands, such as 900 MHz, 2.4 GHz, and 5.8 GHz.

For the UAV-to-UAV channel, we use the Rician model described in [26]. The link characterization has been conducted in [26] using an IEEE 802.11 radio. The received signal of UAV is affected by large scale fading and small scale fading, and path loss was determined by the Friis equation and a fading channel distribution that fits with the height–dependent Rician factor [27]. We assume that the transmission bandwidth of this network is divided into K orthogonal subchannels, where $\mathcal{K} = \{1, 2, \ldots, K\}$. When UAV i transmits signals to UAV j

over subchannel k, the received power at UAV j from UAV i is expressed as $P_{i,j}^{k}(t)$. The rice distribution is defined as

$$p_{\xi}^{k}(d_{i,j}(t)) = \frac{d_{i,j}(t)}{\sigma_0^2} \exp\left(\frac{-d_{i,j}(t)^2 - \rho^2}{2\sigma_0^2}\right) I_0\left(\frac{d_{i,j}(t)\rho}{\sigma_0^2}\right). \tag{1}$$

where $d_{i,j}(t)$ is the transmission distance between UAV i and j, and ρ and σ_0 reflecting the strength of the dominant and non-dominant paths respectively, I_0 is the modified Basel function. In case no dominant path exists ($\rho = 0$), the Rician fading reduces to a Rayleigh fading defined by

$$p_{\xi}^{k}(d_{i,j}(t)) = \frac{d_{i,j}(t)}{\sigma_0^2} \exp\left(\frac{-d_{i,j}(t)^2}{2\sigma_0^2}\right). \tag{2}$$

The received power at UAV j from UAV i is expressed as

$$P_{i,j}^{k}(t) = P_t G(d_{i,j}(t))^{-\alpha} + p_{\xi}^{k}(d_{i,j}(t)). \tag{3}$$

where G is the constant power gain factor introduced into the amplifier and antenna, P_U is the transmit power of a UAV, $d_{i,j}(t)$ is the transmission distance between UAV i and j, and $(d_{i,j}(t))^{-\alpha}$ is the path loss. Finally, the channel model for UAV i, including large-scale and small-scale fading, can be expressed with

$$P_{r[dBm]} = P_{t[dBm]} - p_{\xi[dB]} - L_{[dB]}. \tag{4}$$

where $P_{t[dBm]}$ is the transmit power, determined by transmit power and antenna gain. $L_{[dB]}$ are the large-scale effects, $p_{\xi[dB]}$ are the small-scale effects. The interference from UAV m to UAV j over subchannel k is represented as:

$$I_{m,UAV}^{k}(t) = \psi_{m,k}(t) P_t G(d_{m,j}(t))^{-\alpha}. \tag{5}$$

where $\psi_{m,k}(t)$ is an $N_l \times K$ binary UAV-to-UAV "subchannel pairing matrix" with a value of $\psi_{i,k}(t) = 1$ when subchannel k is assigned to UAV i for UAV-to-UAV transmission; otherwise $\psi_{i,k}(t) = 0$. The SINR at UAV j over subchannel k is shown as:

$$\gamma_{i,j}^{k}(t) = \frac{P_t G(d_{i,j}(t))^{-\alpha} + p_{\xi}^{k}(d_{i,j}(t))}{\sigma^2 + \sum_{m=1,m\neq i}^{N_l} I_{m,UAV}^{k}(t)}. \tag{6}$$

σ^2 is the Gaussian noise variance. Therefore, the rate of data transmission from UAV i to UAV j in subchannel k is:

$$R_{i,j}^{k}(t) = w\log_2\left(1 + \gamma_{i,j}^{k}(t)\right). \tag{7}$$

where w is the channel bandwidth. UAV will choose the link with highest signal-to-noise for data transmission. The method to build the reinforcement learning model is described below.

C. Reinforcement Learning Model Establishment

A Markov decision model was built for the FANET link maintenance problem, which contained five basic elements $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, among which $\mathcal{S}$ is a finite state set, $\mathcal{A}$ is a finite action set, $\mathcal{P}$ is a state transition probability matrix satisfying $\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$, $\mathcal{R}$ is the reward function, and γ is the discount factor satisfying $\gamma \in [0, 1]$.

The key challenge in applying the learning algorithm to the adjustment of the link maintenance cycle lies in how to set the meaning of the above five elements. In the highly dynamic FANET environment, it is hoped that each UAV node can adaptively adjust the link maintenance cycle based on the state of some previous neighbor nodes and the current network status. When the number of neighbor nodes decreases, the node needs to find a communicable link as soon as possible. In contrast, if the number of neighbor nodes and topological changes remain relatively stable, the sending of link maintenance packets can be appropriately reduced. The optimal time cycle, T_H, for each time slot is finally found through a learning algorithm, and this strategy increases the throughput of the whole network without affecting communication. Based on this goal, the specific meanings of these elements are described below.

1) State Space: State $\mathcal{S}$ is the way an agent perceives environment changes. In FANET, the network environment may be complex, and there are many states that can reflect communication characteristics. Therefore, state $\mathcal{S}$ must represent the most critical factor affecting the adjustment of the link maintenance cycle. Commonly used network communication states include link average throughput, end-to-end delay, and queue length of packets to be sent, and some mobility characteristics of UAV can also be used as state inputs, such as node velocity and acceleration. However, each UAV node can learn how to adjust the link maintenance cycle to achieve balanced resource consumption and network performance and, thus, the inputs of the state must be the local information that the node can obtain by itself instead of global variables. In this paper, a definition similar to that described previously [25] was adopted, and the following three variables were used as the state vector $\mathcal{S}$:

- T_s: the time interval between sending two consecutive link maintenance packets;
- T_r: the time interval between receiving two consecutive link maintenance packets from other nodes;
- N_{um}: the number of neighbor nodes.

Both times T_s and T_r are in milliseconds, and T_r can reflect the position information of other nodes relative to this node. If maintenance packets from other nodes have not been received for a long time, the link may be broken due to high-speed movement, and the link maintenance cycle must be shortened; N_{um} reflects its own demand for link maintenance. When the number of neighbors meets the transmission demand, the link maintenance cycle can be extended appropriately. It should be noted that the selected state is directly subject to the influence of the action, i.e., the agent decides the action, a_t , according to the current state, $S_t < T_s, T_r, N_{um} >$. The environment produces the next state, s_{t+1}, under the influence of the action. Obviously, the design proposed in this paper meets these requirements.

2) Action Space: The action, $\mathcal{A}$, means that the agent responds to the changes in state to change the link maintenance cycle. It obtains the optimal link maintenance cycle in each state from historical experience, i.e., the time interval of broadcasting Hello packets. The action can be defined as the specific value of the link maintenance cycle (unit: ms), but this increases the complexity of the action space and affects the convergence performance of the training algorithm. To simplify the operation, the action was set to:

$$T_H^{t+1} = T_H^t + T_a. \tag{8}$$

where the value of T_a is (500, 400, 300, 200, 100, 0, -100, -200, -300, -400, -500), a total of 11 actions (in *ms*). The first type of action is to increase the link maintenance cycle, T_H, at the next moment, which means that the network environment at this moment allows nodes to detect neighbor nodes slowly. There may be two reasons for such decision-making on the part of the agent: the current node has many neighbor nodes for transmission of data packets, or the queue length of the data packets to be sent is short. The second type of action is to reduce the maintenance cycle at the next moment, which means to enhance the rate of link maintenance packets. The third type of action is to maintain T_H consistent with that at the last moment, i.e., the value of T_H is 0.

3) Design of Reward and Value Functions: The value function is defined as the key to establishing a reinforcement learning framework for adjustment of the link maintenance cycle, and represents the direct purpose of the strategy selected by the agent. The routing algorithm proposed in the literature [32] considers the channel level as well as the load capacity, using the data transmission rate to represent the channel quality and the buffer length of the nodes to estimate the load of the network. We also consider the channel quality when defining the value function and load level. In this paper, we define the reward and value functions in four respects.

The first is the channel quality, which is modeled in Section *Channel Model* for UAVs. The goal of the routing algorithm is to select links for which the signal to interference and noise ratio (SINR) is not lower than some predefined threshold.

$$\gamma_{i,j}^k(t) = \frac{P_U(d_{i,j}(t))^{-\alpha}}{\sigma^2 + \sum_{m=1,m\neq i}^{N_l+N_h} I_{m,UAV}^k(t)} \geq \gamma_{\min}^k. \tag{9}$$

where $\gamma_{\min}^k$ represents the minimum satisfactory SINR of the UAV channel, and the link channel level of a node for time slot t satisfies the constraint. Then, the link receives the reward r_{SINR}, defined as:

$$r_{SINR} = \begin{cases} b\log(1+\gamma_{i,j}^k(t)) - v_m p_m(t), & \text{if } \gamma_{i,j}^k(t) \geq \gamma_m^k \\ 0, & \text{otherwise} \end{cases}. \tag{10}$$

where b is the channel bandwidth, $p_m(t)$ is transmission power, and v_m is the discount factor per unit power level. The second part of the reward value is the queue length, n_{send}, of the data packets to be sent by the node—the longer the queue, the greater the punishment, defined as follows:

$$r_{send} = -\log(1 + \mathrm{n}_{send}). \tag{11}$$

where n_{send} is a positive number, which is inversely proportional to the reward value and, therefore, a minus sign is added in front.

Similarly, the degree of changes in the number of neighbors of the node is considered. If the number of neighbor nodes changes frequently compared to the previous period within a period of time, the agent should be punished (i.e., the reward value is negative), while if the number of neighbor nodes remains basically unchanged, it indicates that the network topology is relatively stable at this time and the reward value is positive. Suppose that the reward value is r_{neb}, it can be expressed as:

$$r_{neb} = \begin{cases} a, \text{if } \frac{N_{t+\Delta t}-N_t}{\Delta t} - \frac{N_t - N_{t-\Delta t}}{\Delta t} > \omega \\ b, \text{ if } \frac{N_{t+\Delta t}-N_t}{\Delta t} - \frac{N_t - N_{t-\Delta t}}{\Delta t} < \omega. \end{cases} \tag{12}$$

where N_t is the number of neighbor nodes at time t. Similarly, $N_{t+\Delta t}$ and $N_{t-\Delta t}$ represent the numbers of neighbor nodes at time $t + \Delta t$ and $t - \Delta t$, respectively. ω is the preset value of the degree of variation in the number of neighbor nodes. When the degree of change is greater than ω, the reward value is a, and when the degree of change is less than ω, the reward value is b; It can be inferred: $a \leq 0$, $b > 0$.

The fourth part of the reward value is the overhead for link maintenance, which is reflected in the ratio of the link maintenance packets to the communication data packets sent by the node per unit time, which is defined as:

$$r_{\text{cost}} = -\delta \frac{m_{Hello}}{m_{\text{total}}}. \tag{13}$$

where m_{Hello} is the overhead for link maintenance, .. is the total overhead for sending packets on the channel, and $\delta = \begin{cases} 0, \text{ if } m_{Hello} = m_{\text{total}} \\ 1, \text{ else} \end{cases}$, i.e., when there is no additional demand for sending data packets in the link, the overhead for link maintenance cannot be used as a reward value. Then, the reward value of the node at time t is

$$R_t = \alpha r_{SINR} + \beta r_{send} + \theta r_{neb} + \delta r_{\text{cost}} \tag{14}$$

where α, β, θ and δ are the weights of the four components, respectively, which can be further adjusted according to actual situation. The value function can be defined based on [11]

$$\begin{aligned} Q_\pi(s,a) &\doteq \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a\right] \\ &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a\right]. \end{aligned} \tag{15}$$

where $G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$, and the required strategy $\pi(a|s)$ can be obtained by maximizing the value function:

$$Q_*(s,a) \doteq \max_\pi Q_\pi(s,a). \tag{16}$$

When the FANET requires more links, the UAV decreases the HELLO interval. This causes an increase number of packets for the protocol, and may cause network congestion. However, the design of reward function includes the queue length of packets to be sent and the overhead for link maintenance, which can avoid congestion. When the network load increases with a tendency of congestion, the queue to be sent increases, but the reward function penalizes the agent by prompting it to reduce HELLO packets. So the agent tends to the action that used for reduce link maintenance overhead.

D. Link Maintenance Framework Based on DRL

It can be found from Section *(1) State space* that, although the state defined in this paper contains only three dimensions, i.e., T_s, T_r, and N_{um}, the number of values of each dimension approaches infinity, and it will be impossible to enumerate after

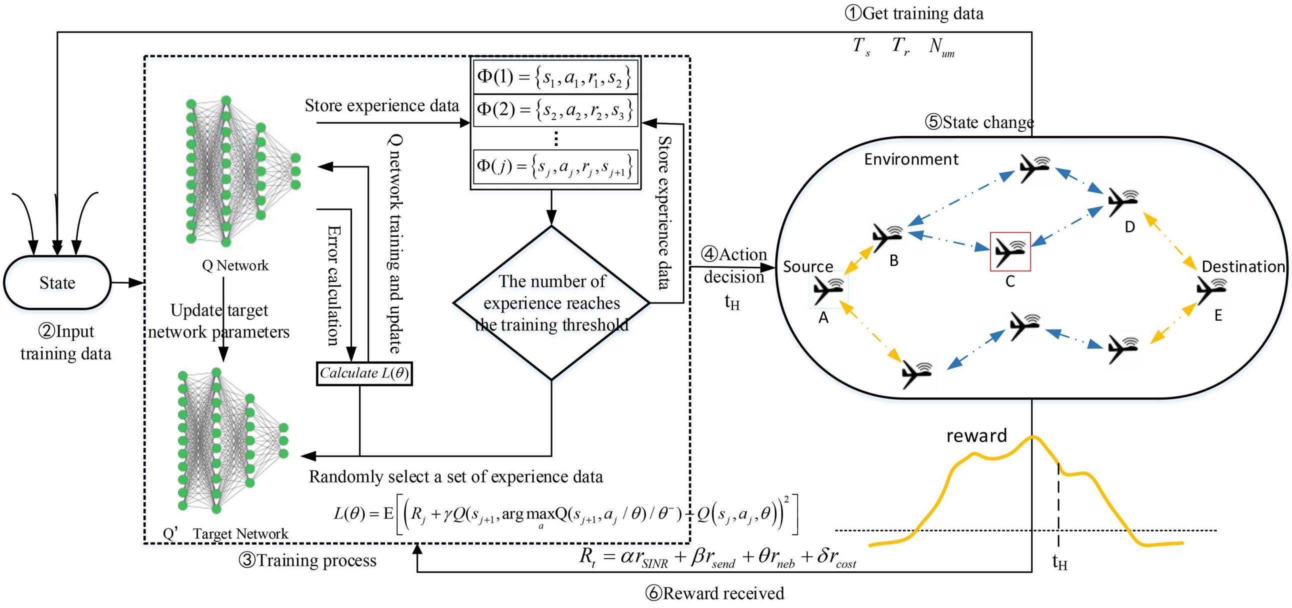

①Get training data
T_s T_r N_{um}
Store experience data
$\Phi(1)=\{s_1,a_1,r_1,s_2\}$
$\Phi(2)=\{s_2,a_2,r_2,s_3\}$
⋮
$\Phi(j)=\{s_j,a_j,r_j,s_{j+1}\}$
Q network training and update
Store experience data
Error calculation
Q Network
State
②Input training data
Update target network parameters
The number of experience reaches the training threshold
Calculate $L(\theta)$
Randomly select a set of experience data
$L(\theta)=\mathrm{E}\left[\left(R_j+\gamma Q(s_{j+1},\arg\max_a Q(s_{j+1},a_j/\theta)/\theta^-)-Q(s_j,a_j,\theta)\right)^2\right]$
Q' Target Network
③Training process
④Action decision
t_H
⑤State change
Environment
Source
A
B
C
D
Destination
E
reward
t_H
$R_t=\alpha r_{SINR}+\beta r_{send}+\theta r_{neb}+\delta r_{cost}$
⑥Reward received

Fig. 4. Framework of the proposed link maintenance system based on DRL.

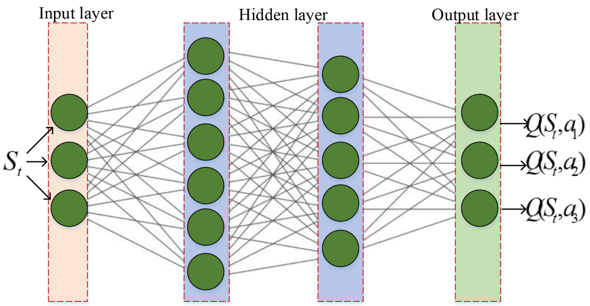


Fig. 5. The principle of approximation of value function.

combination. Therefore, the traditional table-based reinforcement learning cannot solve this problem [28]. In this study, a deep neural network was used to approximate the Q function to solve the above problem and, based on the improvement of the traditional DQN algorithm, a new solution algorithm, DRL-MLsA, suitable for this model was proposed. Fig. 4 shows the framework of the solution proposed in this paper.

Input: The input of the model is the node's state matrix and the determined link maintenance cycle. Specifically, the Fig. 4 shows the time interval of sending maintenance packets, T_s, the time interval of receiving maintenance packets, T_r, the number of neighbor nodes, N_{um}, the action, T_H^{t+1}, and the network output, $Q(s_t, a_t)$:

$$Q(s_t, a_t) = f(V). \tag{17}$$

$$V = \sum_{i=1}^{10} \omega(k+i)y(i). \tag{18}$$

$$y(i) = f(a(i)). \tag{19}$$

$$a(i) = \sum_{j=1}^{n} U(j)\omega(j-1, i). \tag{20}$$

where V is the input of node in the output layer, $\omega(k+i)$ is the weight between the hidden layer and the output layer, $y(i)$ is the output of the hidden node, $a(i)$ is the input of the hidden node, $\omega(j-1, i)$ is the weight between the hidden layer and the input layer, and $f(x) = 1/[1+\exp(-x)]$ is the activation function.

Objective function: As the action value function, Q network extracts state features through a convolutional neural network (CNN), and can map the tensor input of different states into a set of Q values corresponding to different actions. Based on the idea of Q-learning, DQN computes the error between the estimated value of the Q' function of the target network and the Q value predicted by the network using the data of tag delay stored in the experience pool, and further updates the parameters of the target network, such as weight and bias, through stochastic gradient descent to achieve optimization of the action value function and cumulative return. The Q' function of the target network was calculated by the following formula:

$$Q'(s_j, a_j) = R_j + \gamma \max(Q'(s_{j+1}, a_{j+1})). \tag{21}$$

Thus, the loss function is:

$$L(\theta) = \mathrm{E}$$
$$\left[(R_j + \gamma \max(Q'(s_{j+1}, a_{j+1}, \theta')) - Q(s_j, a_j, \theta))^2\right]. \tag{22}$$

To realize the optimization and approximation of the action value function, the error function should be made to approach 0 as much as possible, i.e., the objective function is:

$$\min_\theta \mathrm{E}\left[(R_j + \gamma \max(Q'(s_{j+1}, a_{j+1}, \theta')) - Q(s_j, a_j, \theta))^2\right]. \tag{23}$$

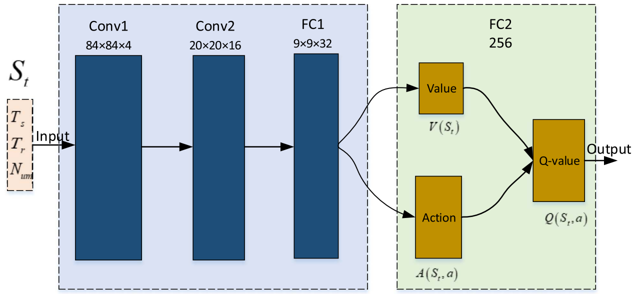


Fig. 6. Link maintenance learning algorithm with dueling architecture.

Training and update: The gradient for updating the weight parameter can be obtained from the loss function:

$$\frac{\partial L_i(\theta_i)}{\partial \theta_i} = E_{(s,a,r,s')-U(D);s'<\varepsilon} \left[y_i - Q(s,a;\theta_i) \frac{\partial Q(s,a|\theta_i)}{\partial \theta_i} \right]. \tag{24}$$

The parameters of the policy network were updated along the direction of gradient descent. A simplified four-layer neural network was used as an example of the Q-value network for description, as shown in the figure, where the input includes the state, s.

Suppose that the state, s, contains two variables, i.e., $S = \{S_1, S_2\}$; the weight between the nodes of each layer is w_{ij}^l, i represents the sequence number of the node in the upper layer, j is the sequence number of the node in the next layer, s_l is the number of nodes in the l-th layer, and l is the sequence number of the weight. For example, w_{31}^1 represents the weight between the third node in the input layer and the first node in the first hidden layer. The bias of each layer in the network is $b_i^{(l)}$. Suppose that the activation functions of the first hidden layer, the second hidden layer, and the output layer in the simplified value network are $f_1(x)$, $f_2(x)$, and $f_3(x)$, respectively, the inputs are z_1, z_2, and z_3, respectively, and the network output is $Q(s_t, a_t)$.

According to the chain rule, the gradient of the weight, w_{ij}^l, of each layer was calculated by calculating the net input and activation value of each layer until the last layer, and then computing the error term of each layer based on backpropagation, and finally calculating the partial derivative of each layer, i.e., the updateable parameter.

IV. DQN-Based Link Maintenance Learning Algorithm-DRL-MLsA

To improve the neural network training effect and speed up convergence, we used a competitive network in the Q network as shown in Fig. 4 to replace the single-output network model in the classic method. As shown in Fig. 6, the action value function, $Q(S_t, a)$, can be naturally divided into two parts: the status value function, $V(S_t)$, and the action advantage function, $A(S_t, a)$. The status value function is not related to the action; the action advantage function is related to the action, which is the quality of the average return of the action relative to the state, and can be used to solve the reward overestimation problem.

The fully connected layer of the classic neural network was divided into an output state function, $V(S_t)$, and an output action advantage function, $A(S_t, a)$, which were finally combined into an action state, $Q(S_t, a)$, through full connection, i.e.,

$$Q(S_t, a) = V(S_t) + A(S_t, a). \tag{25}$$

After the state value function is split, when the action advantage value is fixed, there are infinite possible combinations of the state value and the action advantage value. However, only a small proportion of the combinations are reasonable. As the expected value of the action advantage function $A(S_t, a)$ is 0, this characteristic can be used to constrain the action advantage function, $A(S_t, a)$, and the above formula can be modified as:

$$Q(S_t, a) = V(S_t) + \left(A(S_t, a) - \frac{1}{|A|} \sum_{a'} A(S_t, a') \right). \tag{26}$$

The action advantage function was used to subtract all the mean values of $A(S_t, a')$ in the current state, so that the expected value of the action advantage function can be maintained at 0, thereby ensuring rapid convergence of the model and efficient output.

Stability improvement: the method of updating the value function of Q network is:

$$Q(s_t, a_t) = r_t + \gamma \max_a (Q(s_{t+1}, a_{t+1}; \theta_t)). \tag{27}$$

where $Q(s_{j+1}, a_{j+1}; \theta_t)$ is the neural network's prediction of the value of the state, s_{j+1}, when action, a_j, is adopted. To solve the problem of exploration and utilization effectively, i.e., to try some new actions for greater rewards while at the same time continuing to use the current optimal strategy to maintain high returns, the ε greedy strategy was employed according to the exploration rate, ε:

$$\pi(a/S_t) = \begin{cases} \frac{\varepsilon}{A(s_t)} + 1 - \varepsilon & a^* = S_{t+1}, \max_a Q(S_{r+1}, a; \theta_t) \\ \frac{\varepsilon}{A(s_t)} & \text{else} \end{cases}. \tag{28}$$

where $A(s_t)$ is the action space when the agent is in the state, s_t. After the optimal action, a^* , of the state, s_{j+1}, is selected, the DQN method uses the same parameter, θ_t, to select and evaluate the action. To reduce the influence of the maximum error, another neural network was introduced that uses a different value function to select and evaluate the action. Therefore, the parameter, θ_t, was used to select the action through the above formula and, after the optimal action was chosen, the parameter, θ'_t, of another neural network was adopted to evaluate the action:

$$Q(s_t, a_t) = r_t + \gamma Q(S_{t+1}, a^*; \theta'_t). \tag{29}$$

By applying this idea to the framework, the modified method of updating the value function of the Q network was obtained:

$$Q(s_t, a_t) = r_t + \gamma Q\left(S_{t+1}, \max_a Q(S_{t+1}, a; \theta_t); \theta'_t\right). \tag{30}$$

Algorithm 1: DRL-MLsA DQN-based link maintenance algorithm

```
Input: $S_t < T_s, T_r, N_{um} >$
Output: $Q(s_t, a_t)$
Use random $\theta$ to initialize action value $Q$
Let $\theta_t = \theta$, update the $Q$ value according to (26)
For each scene do:
Initialize the first state and calculate the reward value by (14)
For each step do:
  Use probability, $\varepsilon$, to select action $a$. If the little probability exploration event does not occur, a greedy strategy is used to select the action, $a_t = \max_a Q(s_{t+1}, a; \theta_t)$, with the current maximum action value function
  The environment observes the reward, $r_t$, based on the action, $a_t$
  Set $s_{t+1} = s_t$, integrate $\{s_j, a_j, R_j, s_{j+1}\}$, and store in the playback memory pool
  If empirical data reach the set threshold
    Then
    Update target $Q'$ value through (30)
    Update network parameters through gradient back propagation of the neural network
  End if
Skip to the next step
return: $Q(s_t, a_t)$
```

The loss function is:

$$L(\theta) = E\left[\left(R_j + \gamma Q\left(s_{j+1}, arg\max_a Q(s_{j+1}, a_j/\theta)/\theta'_t\right)\right.\right.$$
$$\left.\left. - \, Q(s_j, a_j, \theta)\right)^2\right]. \tag{31}$$

where

$$\theta'_t = \theta + \alpha\left[R_j + \gamma Q\left(s_{j+1}, \mathrm{argmax}_a\, Q(s_{j+1}, a_j/\theta)/\theta'_t\right)\right.$$
$$\left. - \, Q(s_j, a_j, \theta)\right] \nabla Q(s_j, a_j, \theta). \tag{32}$$

Using algorithm1 to train and solve the DRL-based link maintenance strategy framework, network training can be accelerated so that the agent converges to the optimal link maintenance cycle in each time slot.

As shown in the Fig. 7, in this protocol, the node first senses its own environment, which is given by $S_t < T_s, T_r, N_{um} >$ and contains three dimensions: the time interval between the sending of two consecutive "link maintenance packets", the time interval between the receiving of two consecutive link maintenance packets from other nodes, and the number of neighbors, which is used as a label when obtaining the value function of the corresponding action by DRL-MLSA, to in turn determine whether the Hello interval has increased or decreased. There are 11 actions from which to choose. When the protocol decides to decrease the Hello interval, the node will broadcast Hello messages more frequently because the changes in neighboring nodes are considered by the reward function. Therefore, the network topology changes relatively rapidly at this time and more frequent Hello messages need to be broadcast for link maintenance.

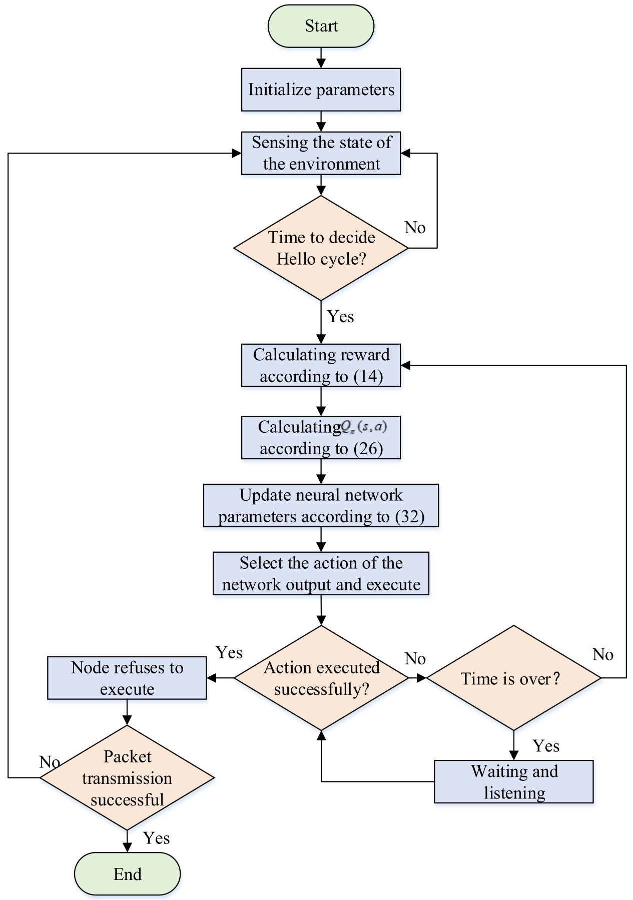

Start
Initialize parameters
Sensing the state of the environment
Time to decide Hello cycle?
No
Yes
Calculating reward according to (14)
Calculating $Q_\pi(s,a)$ according to (26)
Update neural network parameters according to (32)
Select the action of the network output and execute
Yes
Node refuses to execute
Action executed successfully?
No
Time is over?
No
Yes
Waiting and listening
No
Packet transmission successful
Yes
End

Fig. 7. Flowchat of the implementation of DRL-MLSA.

In contrast, when the protocol arrives at the decision to increase the Hello interval, the node slows down the rate at which Hello messages are broadcast, thus reducing the protocol overhead.

V. Performance Evaluation

To verify the feasibility of the proposed solution and analyze the performance of the improved protocol, a simulation was conducted based on the NS3-gym tool [29].

The general structure of the NS3-Gym framework is shown in Fig. 8. Zero Message Queue (ZMQ) is a socket-like message processing queue library that scales elastically across multiple threads, cores, and devices. NS3-Gym implements and encapsulates the underlying functions for data communication between NS3 and OpenAI Gym based on ZMQ, and provides interfaces for information interaction on both sides. Specifically, the NS3 side provides interface functions pertaining to passing status: MyGetObservation(), reward MyGetReward(),end marker MyGetGameOver() and receiving action MyExecuteActions(action). Correspondingly, the OpenAI Gym side provides

TABLE I
COMPARISON OF NETWORK PERFORMANCE BETWEEN OLSR-DRL AND AODV-DRL WITH DIFFERENT TRAINING TIMES

Episodes	Throughput (kbps)		Packet loss rate(%)		Number of Hello packages (pcs)	
	OLSR	AODV	OLSR	AODV	OLSR	AODV
10	17.68	22.34	43.87	42.65	6854	11287
150	19.16	25.06	40.13	40.68	6125	10386
250	20.27	26.79	36.98	38.87	5873	9863
350	22.04	28.46	33.09	36.97	5475	9531
450	22.46	28.89	31.20	35.87	5320	9491
550	22.49	29.03	31.23	35.73	5362	9483

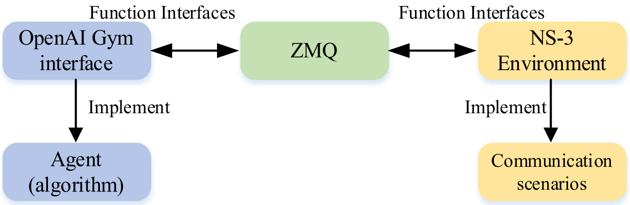


Fig. 8. The framework of NS3-Gym.

the interface functions obs, done, reward = step(action) for receiving status, as well as reward, end flag, and passing action information.

Obtain the dataset: DRL-MLsA learns in the interaction with the environment, and thus a training dataset can be generated by combining the required parameters obtained using the simulation environment, NS3-gym. The statistical state through this interface, such as the state vector, $S_t < T_s, T_r, N_{um} >$, of the node, includes the time interval between sending and receiving two Hello packets in each time slot, and the number of neighbors of the node. At the same time, the reward value and action, $Q(S_t, a)$, at this time were recorded and used as data labels.

Two solutions used for comparison include the traditional solution based on the fixed cycle of broadcasting link maintenance packets and the solution that can adaptively adjust the cycle proposed previously [24]. The performance of the proposed solution was analyzed from the two active routing protocols, i.e., Optimized Link State Routing (OLSR) and AODV. After adjusting the cycle of broadcasting Hello packets using the proposed method in this paper, the superiority of the improved solution was analyzed through testing using performance indicators, such as network throughput, packet loss, and link maintenance overhead. To study the application characteristics of the proposed solution in FANET, the performance of the protocol was verified by changing the movement velocity of the node and the size of the data traffic in the test.

A. Evaluation of Algorithm Convergence

First, the convergence of the proposed training algorithm was analyzed, and statistical analyses were conducted on the loss and reward values of the first 700 training episodes through simulation. A comparison with the performance of the traditional DQN is shown in Fig. 9. The results indicated that the improved DQN algorithm shows faster convergence than the traditional DQN, reaching the optimal value near 300 episodes, and the loss and reward values are basically unchanged with better stability.

To represent visually the performance of the improved algorithm, the performance of the protocol during algorithm iteration was recorded. With the number of UAVs set to 20, RPGM as the mobile model, node velocity of 300 m/s, and network load of 30 kbps, the indicators of the OLSR and ADOV protocols under different numbers of iterations are shown in Table I.

When the number of iterations of the algorithm is less than 10, the performance indicators of the protocol are essentially the same as those of the protocol of the traditional fixed link maintenance cycle. When the number of iterations increases, the throughput of the protocol also rises and the packet loss and number of Hello packets are reduced. When the number of iterations exceeds 500, each performance indicator remains basically unchanged and the algorithm converges.

B. Evaluation of Routing Protocol Performance

The agent obtained from the above training was loaded into the routing protocol, and the UAV only needs to input the corresponding state to obtain the optimal cycle of broadcasting Hello packets without additional iteration calculation. The following simulation results were obtained.

1) The Setting of the Simulation Scene: The performance of the proposed solution was compared and analyzed from two aspects:

a) Through statistical analysis and comparison with the traditional fixed cycle and the solution presented previously [24], statistical analyses were performed on the above-mentioned performance indicators of the two protocols based on the DRL-improved link maintenance mechanism, OLSR and DSDV. The link maintenance framework described in this paper must deal with high-speed changes in UAV topology. To verify the applicability of the algorithm, the speed of UAVs in the simulation is relatively fast, even reaching the speed of sound in an attempt to include all ultra-high-speed aircraft.

b) FANET application scenarios are closely related to application to aircraft. In a previous research [30], the common application scenarios of highly dynamic FANET were divided into three categories, i.e., search, cruise, and target tracking, and the possible movement methods of nodes in these scenarios were also presented. To simulate a complete UAV mission scenario, in the test, the UAV was set to complete these three tasks consecutively, from cruise to target search to target tracking. Based on analysis of the

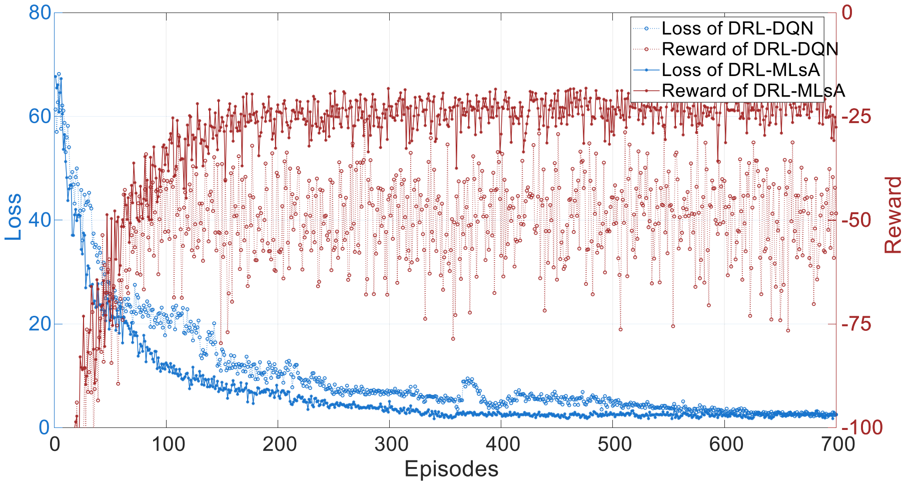


Fig. 9. Convergence performance comparison of the training algorithm.

movement characteristics of these three tasks, the movement model of the UAV node was set to switch between Reference Point Group Mobility (RPGM), Random Way Point (RWP), and Pursue until the end of the simulation. The simulation time of each model was 1000 s, for a total of 3000 s.

In the simulation, we use IEEE 802.11b protocol for transmission. IEEE 802.11b protocol uses Direct Sequence Spread Spectrum (DSSS), which has a total of 14 sub-channels with a bandwidth of 22 MHz available in the 2.4 GHz ISM band, and it provides a maximum transmission speed of 11 Mbps [33], [34]. The DSSS protocol of IEEE 802.11b has the RAKE reception technique, which enables multipath diversity reception, i.e., refracted, reflected, and bypassed signals. RAKE enables higher signal strength, which is beneficial for UAV channels where multipath interference exists. And, Brown [35] used a communication module supporting IEEE 802.11b protocol to measure the signal-to-noise ratio (SINR) and communication capability of the UAV during actual communication, showing that a reliable 1 Mbps rate can be provided at a distance of 10 km in the absence of occlusion. The IEEE 802.11b MAC protocol was chosen to evaluate the proposed routing algorithm under the practical applications. The simulation parameters are shown in Table II.

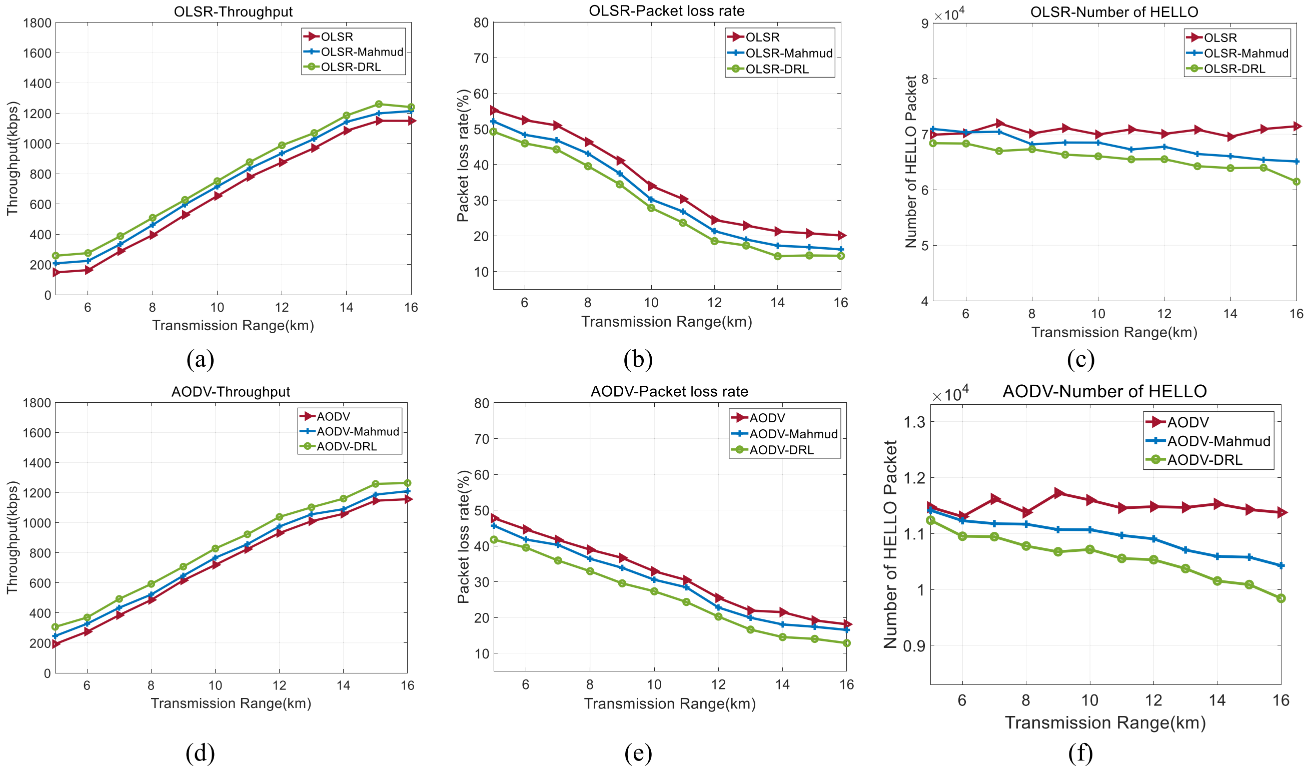
TABLE II
SIMULATION PARAMETERS

Parameter	Value
Simulation area	80km×80km
Transmission range $d_{i,j}(t)$	5-16km
Number of UAV	50
Speed of UAV	50-500 m/s
Simulation time	1000s
Mobile model	RPGM/RWP/Pursue
Routing protocol	OLSR and AODV
Data transmission method	Constant Bit Rate (CBR)
Packet size	512bytes
CBR rate	100k-1.2Mbps
Number of subchannels	10
Transmit power of UAV P_t	9-19 dBm(Antenna Gain 6 dBm)
Power gains factor G	-31.5dB
SINR threshold $\gamma_{\min}^k$	10dB
MAC protocol	IEEE 802.11b

2) Analysis of the Results: Two active routing protocols based on the improved link maintenance cycle were simulated separately, i.e., OLSR and AODV. To verify the impact of UAV flight speed on FANET performance, the velocity of each UAV node was first controlled to perform statistical analysis on the performance of the two protocols.

a) Impact of UAV Communication Range: We consider the SINR of the channel when defining the value function; the communication range of the UAV also affects the performance of the protocol, because the relationship between the signal transmitting power and transmission range is known. We adjust the transmission range of the UAV by changing the transmitting power of the UAV over the range of 5-16 km, and obtain performance statistics. As shown in the Fig. 10, the throughput, packet loss rate, and number of Hello packets of the OLSR protocol are plotted with node speed on the left, and the variation of the AODV protocol is represented on the right [where red is the protocol based on the traditional fixed link maintenance cycle (AODV), blue is the comparison scheme (AODV-Mahmud), and green is the performance of the protocol based on the scheme proposed in this paper (AODV-DRL)]. As shown in Fig. 10(a) and (d), the throughput of both the OLSR and AODV protocols increases with the transmission range. The increase in throughput becomes slower after the transmission range reaches a certain level, indicating that the transmission range is not the only factor determining the performance of the protocol at this point. As shown in Fig. 10(b) and (e), the packet loss rate of both protocols decreases as the transmission range increases. Although the protocol maintains a certain packet loss rate even though the transmission range continues to increase, the method proposed in this paper shows better performance in terms of both the throughput and packet loss rate.

We achieve different transmission range in NS3 by adjusting the signal transmitting power of the UAV, the higher the signal transmitting power, the longer the transmission range. When



(a) (b) (c)

(d) (e) (f)

Fig. 10. Impact of UAV communication range on network performance.

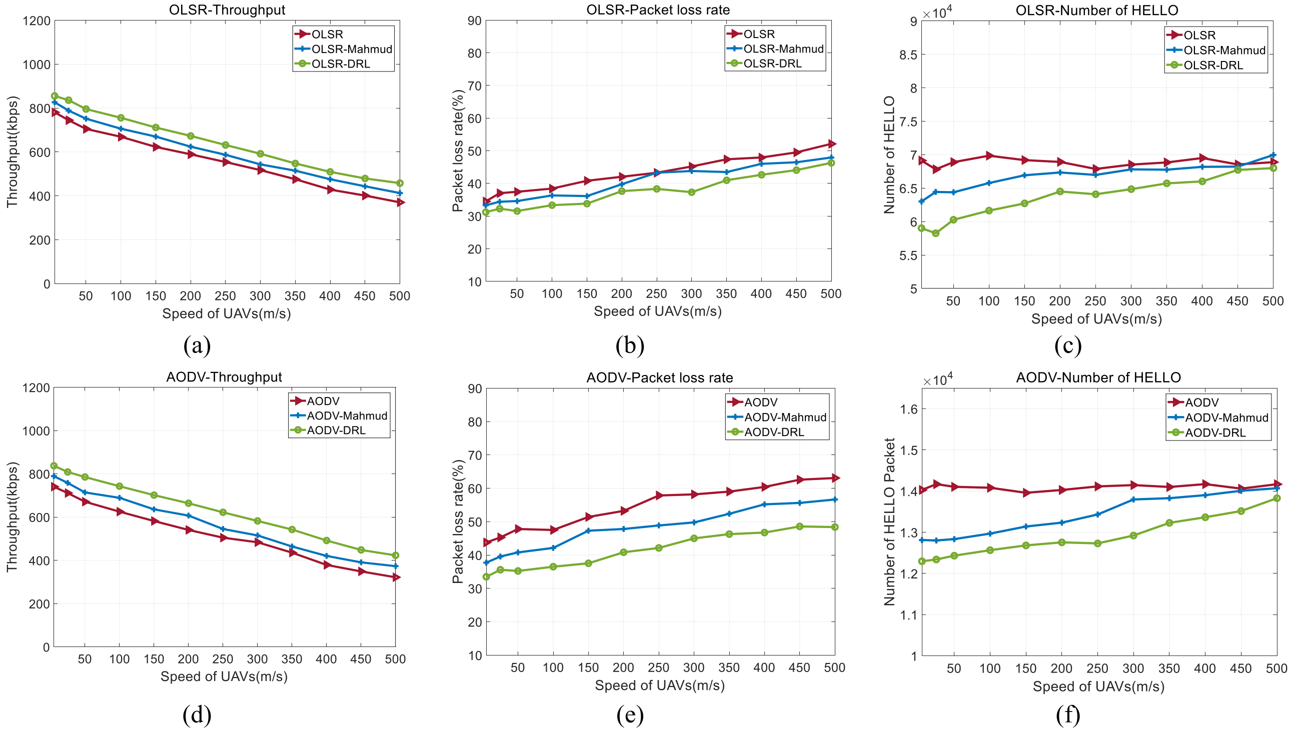
drawing Fig. 10, the horizontal coordinate is the transmission range, while it can actually be considered as the transmit power of the node, since the higher the transmit power, the greater the transmission range of the node. Obviously, as the transmit signal is enhanced, the probability of packet loss decreases. That is, the longer the transmission range, the smaller the packet loss rate.

Both protocols, when not optimized, broadcast Hello packets within a fixed period for link maintenance. The total number of Hello packets does not vary with the transmission range, as shown in Fig. 10(c) and (f). The algorithm in the comparison scheme considers the speed and transmission range of the UAV. It can be seen from the experimental results that, using that algorithm, the number of Hello packets decreases with increasing transmission range, resulting in an improvement of energy efficiency. The method described in this paper also establishes the channel model of the UAV and considers SINR when defining the value function. The number of Hello packets decreases with increasing transmission range, which reduces the protocol overhead in a highly dynamic environment.

b) Impact of Node Speed: In the Fig. 11(a) and (d), the indicator diagram of the changes in the throughput, packet loss, and the number of Hello packets of the OLSR protocol with the node velocity is shown on the above. The variation of the AODV protocol is shown on the bottom. The protocol based on the traditional fixed link maintenance cycle (AODV) is indicated in red, the comparison solution (AODV-Mahmud) is indicated in blue, and the performance of the protocol based on the solution proposed in this paper (AODV-DRL) is indicated in green.

In terms of throughput, as shown in Fig. 11(b) and (e), the throughput of both OLSR and AODV protocols decreased with increasing UAV speed, which was consistent with the actual situation. Compared to the protocol based on a fixed link maintenance cycle, the protocol based on the improved solution enhanced the throughput of the whole network at various speeds. In contrast, as shown in Fig. 1(c) and (d), packet loss increased with increasing speed of UAV. However, packet loss of the protocol based on the improved solution was reduced, although the packet loss performance was inferior to that of the comparison solution at some speeds.

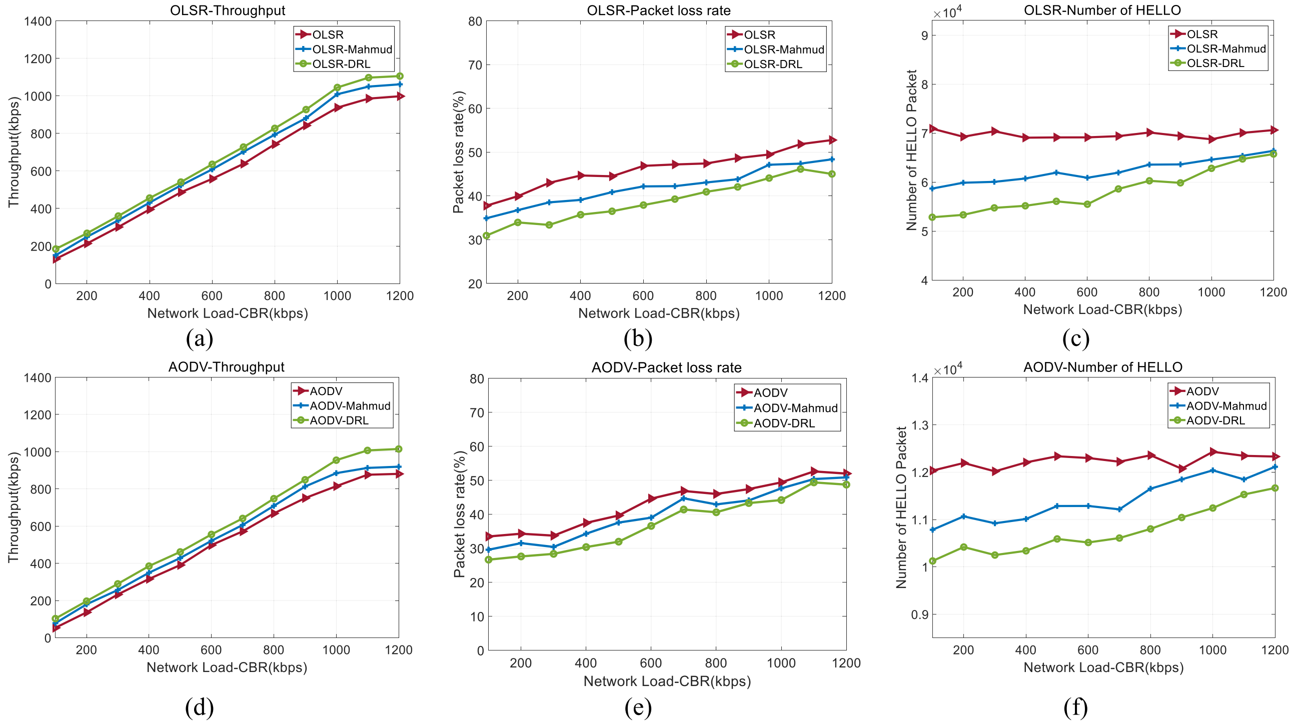
In terms of the number of Hello packets, as shown in Fig. 11(c) and (f), given a fixed link maintenance cycle, the number of Hello packets of the two protocols will not change with the speed of UAV. For example, the number of Hello packets of the OLSR protocol was maintained at about 68000 (simulation time: 1000 s). When the UAV speed was low, the protocol of the comparison solution had fewer Hello packets than the fixed cycle-based protocol, which conforms to the adaptive Hello interval algorithm that considers the speed of UAV in this paper, and also achieves the author's purpose, i.e., to reduce the energy loss of link maintenance in a highly dynamic UAV environment. When the UAV speed was low, the number of Hello packets of the protocol based on the Hello interval solution proposed in this paper was relatively small, and increased with increasing speed. This is because the change rate of UAV's neighbor nodes and the link maintenance overhead are considered in the reward and value functions. When the speed is low, the changes in UAV's neighbor nodes are relatively stable. The agent can only acquire a positive reward value by increasing the time interval between Hello packets, otherwise it will be punished.



(a) (b) (c)

(d) (e) (f)

Fig. 11. Impact of UAV speed on network performance.



(a) (b) (c)

(d) (e) (f)

Fig. 12. Impact of service load on network performance.

c) Impact of Network Load: Network load is the size of data traffic generated by service demands in FANET. We set the CBR rate to 100kbps-1.2Mbps to verify the performance of the algorithm and to make the simulation experiments closer to the facts. The simulation generates CBR data streams of 512bytes in size and 4-48 in number per second, and the number of UAVs is 50. In the Fig. 12, the indicator diagram of the changes in the throughput, packet loss, and the number of Hello packets of the OLSR protocol with the size of the network load is shown on the above. The variation of the AODV protocol is shown on the bottom. In terms of the throughput, as shown in Fig. 12(a) and (d), the throughput of the two protocols increased as the network load rose. The throughput of the OLSR or AODV protocol based on the improved link maintenance cycle was greater than those for the protocols of other solutions.

As shown in Fig. 12(b) and (e), the packet loss of the two protocols increased with increasing network load. This was because when the network environment cannot meet the requirements of load, more and more data packets will be lost in the links, which conforms to the actual situation. Generally, the packet loss of the protocol based on the improved link maintenance cycle was smaller than that of the fixed cycle-based protocol because the maintenance overhead was considered in the value function. The agent can adaptively balance the consumption of link maintenance and the consumption of data transmission, and appropriately reduce the maintenance overhead to increase the link resource space for data transmission, thereby decreasing packet loss of the network.

Viewed from the number of Hello packets, as shown in Fig. 12(c) and (f), the number of Hello packets of the protocol based on the fixed link maintenance cycle did not change with the network load. For example, the number of Hello packets of the OLSR protocol was basically maintained at about 69000 while the modified protocol of the comparison solution had a small number of Hello packets under a small load. This is because the task-related traffic factors are considered in the adaptive Hello interval solution in this paper, and the number of Hello packets will increase with increasing network load; the protocol based on the Hello interval solution proposed in this paper has few Hello packets under small network load, and the number will increase as the load increases because the queue length of the data packets to be sent and the link maintenance overhead are taken into account in the definition of the reward and value functions. When the network load is small, the queue length of the data packets to be sent is small, and the agent will obtain a positive reward by increasing the interval between Hello packets.

3) A timing simulation was carried out to simulate a practical application scenario of UAV. Fig. 13 shows the simulation results of the performance of the AODV routing protocol based on the traditional fixed link maintenance cycle, and based on the improved Hello interval proposed in this paper. The throughput of the protocol based on the improved method was generally higher than that of the protocol based on the traditional fixed cycle (CBR = 300kbps). In addition, the throughput was reduced between 1000 s and 2000 s, indirectly reflecting that the routing protocol is affected by the UAV mobility model. The network performance of the AODV protocol under the RWP mobility model was poorer than those under other mobility models; this was also verified by packet loss analysis. As shown in the second column of the Fig. 13, the packet loss in the middle was slightly higher than at both sides. The number of Hello packets increased with increasing simulation time, but the number of Hello packets of the protocol based on the improved solution was about 5000 less than that of the fixed cycle-based protocol.

Finally, the network performance of the three protocols is summarized in Table III. The statistical parameters are based on the average calculation after multiple tests. The protocol in the first line is the original protocol based on the fixed link maintenance cycle, and the value in brackets is the performance improvement rate compared to the original protocol. The DRL solution reduced the number of data packets used by nodes to maintain links, thereby improving the throughput of the whole network, and reducing the packet loss accordingly.

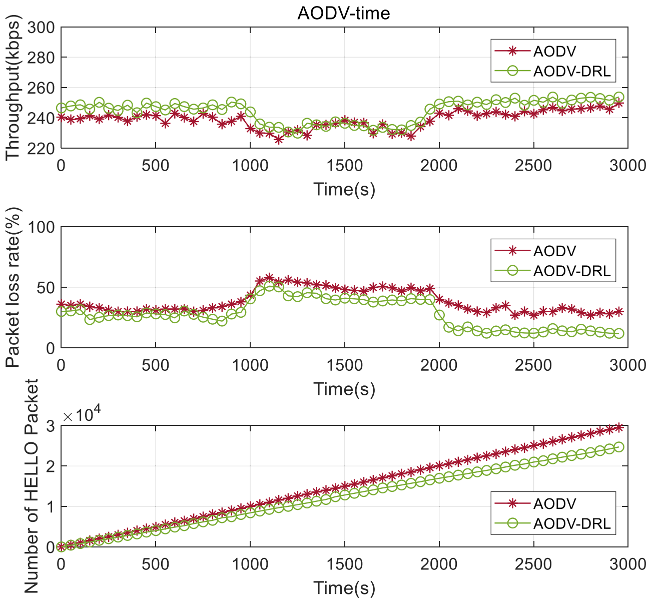


Fig. 13 AODV-DRL protocol performance results over time.

TABLE III
SIMULATION RESULTS

Protocol	Throughput (kbps)	Packet loss rate (%)	Number of Hello packages (pcs)
OLSR	169.54	37.02	10092
OLSR-Mahmud	181.36 (+6.97%)	34.17 (-7.69%)	8969 (-11.13%)
OLSR-DRL	198.96 (+17.35%)	29.57 (-20.13%)	7995 (-20.78%)
AODV	184.64	36.12	13527
AODV-Mahmud	216.29 (+17.14%)	34.04 (-5.75%)	12117 (-10.42%)
AODV-DRL	226.24 (+22.53%)	31.01 (-14.15%)	11609 (-14.18%)

VI. CONCLUSION

In this paper, we proposed a scheme for link maintenance in highly dynamic FANET, which can be applied to all active routing protocols. The mobility parameters of the UAV in FANET was used to calculate the link maintenance strategy and adaptively adjust the period of broadcast Hello packets. The algorithm took multiple goals such as increasing the network throughput of the FANET, reducing link maintenance overhead, and transmission packet loss rate as the optimization goals. To solve this joint optimization problem, we used the reinforcement learning framework to establish the highly dynamic FANET link maintenance problem as a Markov decision process, learn the corresponding link strategy by sensing the network state characteristics brought by UAV movement, and propose the DRL-MLsA algorithm to improve the stability and speed of the training agent. Our experiments showed that the OLSR and AODV routing protocols based on the proposed framework had better network performance than traditional protocols.

Acknowledgment

The authors would like to thank the Associate Editor and the anonymous reviewers for their constructive comments, which helped them improve the presentation of the work considerably.

References

[1] J. A. Gonçalves and R. Henriques, "UAV photogrammetry for topographic monitoring of coastal areas," *ISPRS J. Photogrammetry Remote Sens.*, vol. 104, pp. 101–111, 2015.

[2] F. Xiong, A. Li, H. Wang, and L. Tang, "An SDN-MQTT based communication system for battlefield UAV swarms," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 41–47, Aug. 2019, doi: 10.1109/MCOM.2019.1900291.

[3] N. I. Mowla, N. H. Tran, I. Doh, and K. Chae, "AFRL: Adaptive federated reinforcement learning for intelligent jamming defense in FANET," *J. Commun. Netw.*, vol. 22, no. 3, pp. 244–258, Jun. 2020, doi: 10.1109/JCN.2020.000015.

[4] S. Sankarasrinivasan et al., "Health monitoring of civil structures with integrated UAV and image processing system," *Procedia Comput. Sci.*, vol. 54, pp. 508–515, 2015.

[5] B. Wang, Y. Sun, N. Zhao, and G. Gui, "Learn to coloring: Fast response to perturbation in UAV-Assisted disaster relief networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3505–3509, Mar. 2020, doi: 10.1109/TVT.2020.2967124.

[6] A. Guillen-Perez and M.-D. Cano, "Flying ad hoc networks: A new domain for network communications," *Sensors*, vol. 18, no. 10, 2018, Art. no. 3571.

[7] I. Bekmezci, O. K. Sahingoz, and Ş. Temel, "Flying ad-hoc networks (FANETs): A survey," *Ad Hoc Netw.*, vol. 11, no. 3, pp. 1254–1270, 2013.

[8] D. Shumeye Lakew, U. Sa'ad, N. Dao, W. Na, and S. Cho, "Routing in flying ad hoc networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1071–1120, Apr.–Jun. 2020, doi: 10.1109/COMST.2020.2982452.

[9] S. Y. Han and D. Lee, "An adaptive hello messaging scheme for neighbor discovery in on-demand MANET routing protocols," *IEEE Commun. Lett.*, vol. 17, no. 5, pp. 1040–1043, May 2013, doi: 10.1109/LCOMM.2013.040213.130076.

[10] R. Oliveira, M. Luis, L. Bernardo, R. Dinis, and P. Pinto, "The impact of node's mobility on link-detection based on routing hello messages," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2010, pp. 1–6, doi: 10.1109/WCNC.2010.5506529.

[11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[12] A. Valadarsky et al., "Learning to route with deep RL," in *Proc. NIPS Deep Reinforcement Learn. Symp.*, 2017, pp. 1–11.

[13] D. Marconett et al., "Self-adapting protocol tuning for multi-hop wireless networks using Q-learning," *Int. J. Netw. Manage.*, vol. 23, no. 2, pp. 119–136, 2013.

[14] T. Safdar, H. B. Hasbulah, and M. Rehan, "Effect of reinforcement learning on routing of cognitive radio ad-hoc networks," in *Proc. IEEE Int. Symp. Math. Sci. Comput. Res.*, 2015, pp. 42–48, doi: 10.1109/ISMSC.2015.7594025.

[15] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[16] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11158–11168, Nov. 2019, doi: 10.1109/TVT.2019.2935450.

[17] H. Ye, G. Y. Li, and B. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3163–3173, Apr. 2019, doi: 10.1109/TVT.2019.2897134.

[18] Y. Dai, D. Xu, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4312–4324, Apr. 2020, doi: 10.1109/TVT.2020.2973705.

[19] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10190–10203, Nov. 2018, doi: 10.1109/TVT.2018.2867191.

[20] T. Wu et al., "Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8243–8256, Aug. 2020, doi: 10.1109/TVT.2020.2997896.

[21] N. Hernandez-Cons, S. Kasahara, and Y. Takahashi, "Dynamic hello/timeout timer adjustment in routing protocols for reducing overhead in MANETs," *Comput. Commun.*, vol. 33, no. 15, pp. 1864–1878, 2010.

[22] S. Y. Han and D. Lee, "An adaptive hello messaging scheme for neighbor discovery in on-demand MANET routing protocols," *IEEE Commun. Lett.*, vol. 17, no. 5, pp. 1040–1043, May 2013, doi: 10.1109/LCOMM.2013.040213.130076.

[23] V. C. Giruka and M. Singhal, "Hello protocols for ad-hoc networks: Overhead and accuracy tradeoffs," in *Proc. IEEE 6th Int. Symp. World Wireless Mobile Multimedia Netw.*, 2005, pp. 354–361, doi: 10.1109/WOWMOM.2005.50.

[24] I. Mahmud and Y. Cho, "Adaptive hello interval in FANET routing protocols for green UAVs," *IEEE Access*, vol. 7, pp. 63004–63015, 2019, doi: 10.1109/ACCESS.2019.2917075.

[25] W. Li, F. Zhou, K. R. Chowdhury, and W. Meleis, "QTCP: Adaptive congestion control with reinforcement learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 445–458, Jul.–Sep. 2019, doi: 10.1109/TNSE.2018.2835758.

[26] N. Goddemeier and C. Wietfeld, "Investigation of air-to-air channel characteristics and a UAV specific extension to the rice model," in *Proc. IEEE Glob. Commun. Conf.*, 2015, pp. 1–5.

[27] M. Sbeiti, N. Goddemeier, D. Behnke, and C. Wietfeld, "PASER: Secure and efficient routing approach for airborne mesh networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 3, pp. 1950–1964, Mar. 2016.

[28] C. Wu and Y. Wang, "Learning from big data: A survey and evaluation of approximation technologies for large-scale reinforcement learning," in *Proc. IEEE Int. Conf. Comput. Inf. Technol.*, 2017, pp. 1–8, doi: 10.1109/CIT.2017.11.

[29] P. Gawłowicz and A. Zubow, "ns3-gym: Extending openai gym for networking research," 2018, *arXiv:1810.03943*.

[30] J. Hong and D. Zhang, "TARCS: A topology change aware-based routing protocol choosing scheme of FANETs[J]," *Electronics*, vol. 8, no. 3, pp. 274–293, 2019.

[31] X. Cheng et al., "UAV communication channel measurement, modeling, and application," *J. Commun. Inf. Netw.*, vol. 4, no. 4, pp. 32–43, 2019.

[32] A. Al-Saadi, R. Setchi, Y. Hicks, and S. M. Allen, "Routing protocol for heterogeneous wireless mesh networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9773–9786, Dec. 2016.

[33] C. Dixon and E. W. Frew, "Optimizing cascaded chains of unmanned aircraft acting as communication relays," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 5, pp. 883–898, Jun. 2012.

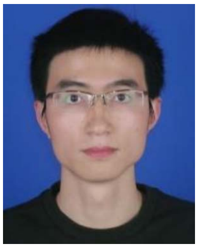
[34] M. Y. Arafat and S. Moh, "Location-aided delay tolerant routing protocol in UAV networks for post-disaster operation," *IEEE Access*, vol. 6, pp. 59891–59906, 2018.

[35] T. X. Brown, B. Argrow, C. Dixon, S. Doshi, R. G. Thekkekunnel, and D. Henkel, "Ad hoc UAV ground network (AUGNet)," in *Proc. AIAA Unmanned Unlimited Tech. Conf.*, Sep. 2004, pp. 29–39.

Xiulin Qiu (Student Member, IEEE) was born in Ganzhou, Jiangxi Province. He received the master's degree. He is currently working toward the Ph.D. degree with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His research interests include deep-reinforcement learning, resource allocation for 5G, and artificial intelligence-based future mobile network.

Yuwang Yang received the B.S. degree from Northwestern Polytechnical University, Xi'an, China, in 1988, the M.S. degree from the University of Science and Technology of China, Hefei, China, in 1991, and the Ph.D. degree from the Nanjing University of Science and Technology (NUST), Nanjing , China, in 1996. He is currently a Professor with the School of Computer Science and Engineering, NUST. His research interests include high-performance computing, machine learning, and intelligent system.

Lei Xu (Member, IEEE) received the bachelor's, master's, and Ph.D. degrees in communication and information system from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2006, 2009, and 2012, respectively. He is currently a Full Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology. He has authored or coauthored more than 50 journal papers, e.g., IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. His research interests include network analysis and resource allocation for 5G and 6G.



Jun Yin received the Ph.D. degree in computer science and technology from the Nanjing University of Science and Technology, Nanjing, China, in 2017. He is currently with the Jiangsu Key Laboratory for Broadband Wireless Communication and Internet of Things, Nanjing University of Post and Telecommunications, Nanjing. His research focuses on the theory and applications of network coding.

Zhenqiang Liao received the Ph.D. degree in mechanical engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1987. He is currently working with the School of Electrical and Mechanical Engineering, Suzhou Global Institute of Software Technology, Suzhou, China. His research interests include UAV design theory and control methods.