Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

WDQAR:A Weighted Double Q-Learning Based Adaptive Routing Protocol for Flying Ad Hoc Networks

Qingzhong Ni

nqingzhong@163.com

PLA Information Engineering University

Yijie Bai

PLA Information Engineering University

Daojie Yu

PLA Information Engineering University

Qi Zhang

PLA Information Engineering University

Fan Meng

PLA Information Engineering University

Article

Keywords: Flying ad hoc networks(FANETs), Routing protocol, Reinforcement learning(RL), Unmanned aerial vehicles(UAVs)

Posted Date: September 15th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10530890/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

WDQAR:A Weighted Double Q-Learning Based Adaptive Routing Protocol for Flying Ad Hoc Networks

Qingzhong Ni[1*], Yijie Bai[1], Daojie Yu[1], Qi Zhang[1], Fan Meng[1]

[1*]School of Information Systems Engineering, People's Liberation Army Information Engineering University, Science Avenue, Zhengzhou, 450001, Henan, China.

*Corresponding author(s). E-mail(s): nqingzhong@163.com;
Contributing authors: byijie0117@163.com; ydj2008@126.com;
qzhang0706@163.com; mengfan.1993@163.com;

Abstract

Reliable routing in flying ad hoc networks (FANETs) faces challenges such as frequent topology changes and unstable links, which are caused by the rapid mobility of UAVs. Existing Q-learning-based protocols still suffer from biased Q-value estimation, slow convergence, and high control overhead. This paper proposes WDQAR, an adaptive routing protocol based on weighted double Q-learning. WDQAR consists of two stages: neighbor discovery and routing. In the neighbor discovery phase, each UAV dynamically adjusts the HELLO message broadcast interval based on link duration and speed, promptly perceiving topology changes and reducing control overhead. In the routing phase, WDQAR applies a sector-based candidate neighbor filtering mechanism to narrow down the next-hop candidate set, thereby accelerating the selection of the optimal path. The reward function comprehensively considers residual communication energy, link duration, geographical progress, and valid neighbor ratio. Then, a weighted double Q-learning framework is adopted during Q-value update to mitigate the estimation bias of Q-learning towards future values, thereby enhancing the stability of routing strategies.In addition, the protocol adopts two optimization methods, namely Q-value initialization and adaptive learning parameters, to accelerate the learning process.Simulation results based on NS-3 demonstrate that WDQAR outperforms the comparative protocols in terms of packet delivery rate, end-to-end delay, energy consumption, and routing overhead.

Keywords: Flying ad hoc networks(FANETs), Routing protocol, Reinforcement learning(RL), Unmanned aerial vehicles(UAVs)

1 Introduction

In recent years, unmanned aerial vehicles (UAVs) have been widely deployed in both civil and military domains, including agricultural production [1], environmental monitoring [2], disaster relief [3], battlefield reconnaissance and strike operations [4], owing to their high mobility and low deployment cost.Within the communication architecture of multi-UAV systems, Flying Ad Hoc Networks (FANETs) enhance the cooperative capability of UAV swarms by enabling efficient information exchange among nodes [5].FANETs adopt a decentralized architecture and do not require centralized access to infrastructure. UAVs are connected to each other in a self-organizing manner.As a result, FANETs offer distinct advantages in areas with poor conditions and limited communication infrastructure [6, 7].

In resource-constrained and highly dynamic networks, ensuring continuous and reliable data transmission remains a core challenge for FANETs. The design of effective and reliable routing mechanisms has therefore become a key research priority [8].However, FANETs have the characteristics of fast node movement, frequent topology changes, and uneven node distribution.Therefore, conventional routing protocols developed for Mobile Ad Hoc Networks (MANETs) and Vehicular Ad Hoc Networks (VANETs) struggle to meet the requirements of FANETs [9].In addition,FANETs face further inherent constraints,including limited battery life, restricted onboard computing resources, and intermittent connectivity caused by sparse node deployment [10].

To address these challenges, extensive research has focused on enhancing conventional routing protocols [11–14]. FANET routing protocols are generally classified into two main categories: topology-based protocols and location-based protocols. Topology-based routing protocols determine data packet transmission paths based on the network's topological connectivity information and ensure reliable packet delivery through dynamic routing table maintenance. However, the high mobility of UAVs in FANETs causes frequent and rapid changes in network topology, resulting in substantial control overhead during route reconstruction and significantly degrading overall network performance[15].Location-based routing protocols utilize node location information to assess neighbor states and select the optimal next hop from candidate neighbors for packet forwarding.These protocols do not need to maintain a global link-state table, thereby reducing routing overhead and improving scalability[16].

However, location-based routing protocols mainly rely on geographic information for next-hop selection, making them sensitive to location inaccuracies and prone to local optimum problems[17, 18]. Moreover, they usually lack the ability to learn from dynamic network conditions, which limits their adaptability in highly mobile FANET environments[19, 20].

In recent years, machine learning has provided new solutions to the routing protocol problems of FANETs owing to its adaptive learning capability. Among them,

Q-learning is a classic model-free reinforcement learning algorithm[21]. It continuously updates the Q-value table through interactions between agents and the environment based on a reward-and-punishment mechanism, thereby dynamically optimizing routing strategies without requiring prior data.Q-learning-based routing protocols can dynamically adjust routing strategies according to real-time network states, enabling them to effectively adapt to the highly dynamic and unstable topology of FANETs.

In this paper, we propose an adaptive routing strategy based on weighted double Q-learning, named WDQAR. WDQAR consists of two phases: the neighbor discovery phase and the routing phase. In the neighbor discovery phase, the broadcast interval of HELLO messages is adaptively adjusted according to the motion and link states of nodes, ensuring that topology changes can be detected in a timely manner. The routing phase adopts a reinforcement learning framework based on weighted double Q-learning. This framework can effectively reduce the estimation bias of the value function, improve the stability of the routing strategy. In addition, the convergence is accelerated by initializing the Q-values and adjusting the adaptive parameters, enabling nodes to learn the routing strategy adapting to the current network environment more quickly. The main contributions of this method are as follows:

- An adaptive HELLO message broadcast interval adjustment mechanism is proposed. UAVs maintain neighbor relationships by periodically broadcasting HELLO messages. The transmission frequency of HELLO messages is dynamically adjusted based on link duration and node velocity, thereby ensuring timely neighbor table updates and reducing control overhead.
- A distributed routing protocol based on weighted double Q-learning is proposed.By maintaining two Q-tables and using an adaptively weighted future estimate, the proposed method reduces Q-value overestimation and improves routing stability in highly dynamic FANETs.
- In WDQAR, the reward function is formulated based on four metrics: residual energy, link duration, geographic progress, and valid-neighbor ratio. Information entropy is further introduced to dynamically determine the weights of these reward components.
- Two optimization strategies are proposed: Q-value initialization and adaptive adjustment of the learning parameters.The former reduces ineffective exploration in the early learning phase and accelerates convergence, while the latter adapts the learning behavior according to network dynamics, improving the adaptability of the routing policy to highly dynamic FANET environments.

2 Related work

The high mobility of FANET nodes causes rapid and unpredictable topology changes, making it difficult for conventional routing protocols to adapt in time. Reinforcement learning enables nodes to learn from environmental feedback and make adaptive forwarding decisions without requiring global network knowledge, making it suitable for routing in dynamic FANETs.

In [22], a Q-learning-based geographic routing protocol called QGeo is presented for unmanned robotic networks in high-mobility environments. QGeo improves next-hop selection through a packet-travel-speed-based reward function that jointly considers forwarding progress, link error rate, and location error, and it introduces a dynamic discount factor based on the predicted neighbor position to reflect link instability. However, QGeo is evaluated only in a small two-dimensional scenario and relies on conventional Q-learning with a relatively simple reward design, which limits its effectiveness in the large-scale, highly dynamic conditions of FANETs.

In [23], a Q-learning-based multi-objective routing protocol called QMR is presented for FANETs, which jointly optimizes end-to-end delay and residual energy in its reward function. QMR adaptively adjusts the learning rate and discount factor according to one-hop delay and neighbor mobility, respectively, and adopts a network-condition-driven exploration?exploitation mechanism. Although QMR raises the packet arrival ratio while reducing delay and energy consumption, its HELLO adjustment relies mainly on node speed and gives limited consideration to link lifetime and topology variation.

In [24], an improved Q-learning-based routing protocol called Q-FANET is presented for FANETs, combining and adapting features from QMR and Q-Noise+. By exploiting SINR and weighted historical episodes, Q-FANET lowers end-to-end delay and jitter while slightly improving the packet delivery ratio. However, it does not consider energy consumption and uses a fixed learning rate and discount factor, which limits its performance in energy-constrained and highly dynamic FANET scenarios.

In [25], a Q-learning-based topology-aware routing protocol called QTAR is presented for FANETs. QTAR uses two-hop neighbor information to extend the local topology view, adjusts the HELLO interval and link-holding time according to link duration, and designs a reward function that accounts for delay, forwarding velocity, and residual energy. Simulations show that QTAR improves the packet delivery ratio, reduces end-to-end delay, and extends network lifetime. However, its reliance on two-hop neighbor information increases routing overhead and system complexity.

In [26], a topology-aware resilient routing protocol based on adaptive Q-learning, called TARRAQ, is proposed for FANETs. TARRAQ models dynamic topology changes with queuing theory, deriving the neighbor change rate and change inter-arrival-time distribution to adapt the sensing interval to network dynamics and traffic demand, and it predicts the residual link duration via Kalman filtering to keep neighbor information valid. Its reward function incorporates link quality, a neighbor metric, and distance to the destination, while the action-selection strategy, learning rate, and discount factor are all adjusted according to the residual link duration. Although TARRAQ improves the packet delivery ratio and reduces routing overhead and energy consumption, its end-to-end delay is slightly higher than that of QTAR because it prioritizes link stability and energy efficiency.

In [27], a Q-learning-based routing scheme with an intelligent filtering algorithm, called QRF, is proposed for FANETs. QRF filters candidate next-hop UAVs within a destination-oriented forwarding region to shrink the state space and accelerate convergence, incorporates connection time, residual energy, and link quality into its reward function, and adaptively tunes the learning parameters to enhance routing

stability. Although QRF reduces delay, balances energy consumption, and extends network lifetime, its filtering mechanism may over-restrict the candidate set in sparse or highly dynamic topologies, reducing routing flexibility and increasing the risk of route-discovery failure.Table 1 provides a comparison of WDQAR with other existing protocols based on Q-learning.

Table 1 Comparison of representative routing protocols based on Q-learning.

Protocol	Adaptive HELLO	Residual energy	Link status	Neighbor filter	Forward progress	Valid neighbor ratio	Q-value init
QGeo	×	×	✓	×	✓	×	×
QMR	✓	✓	×	✓	×	×	×
Q-FANET	×	×	✓	✓	×	×	×
QTAR	✓	✓	✓	✓	✓	×	×
TARRAQ	✓	×	✓	×	✓	✓	×
QRF	✓	✓	✓	✓	×	×	×
WDQAR	✓	✓	✓	✓	✓	✓	✓

3 Basic concepts

3.1 Q-learning

Reinforcement learning learns a decision-making policy through agent-environment interaction, and is commonly formulated as a Markov decision process (MDP). An MDP is defined by the tuple (S, A, P, R, γ), where S denotes the state set, A denotes the action set, $P(s'|s,a)$ represents the transition probability from state s to state s' after executing action a, $R(s,a)$ denotes the immediate reward obtained when action a is performed in state s, and $\gamma \in [0,1)$ is the discount factor. At each time step, the agent observes the current state, selects an action according to its policy, and receives an immediate reward while the environment transitions to a new state. Q-learning is a model-free reinforcement learning algorithm that learns the action-value function without prior knowledge of the state transition model. By iteratively updating the Q-value of each state-action pair, the agent estimates the expected long-term return of different actions and selects the optimal one accordingly. The Q-value update formula at time step t for agent i is as follows[28]:

$$\begin{aligned} Q_i^{t+1}\left(s_i^t, a_i^t\right) = (1-\alpha) Q_i^t\left(s_i^t, a_i^t\right) \\ + \alpha\left\{r^{t+1}\left(s_j^{t+1}\right) + \gamma V_j^t\left(s_j^{t+1}\right)\right\}, \end{aligned} \tag{1}$$

where $\alpha \in (0,1]$ is the learning rate, $\gamma \in [0,1)$ is the discount factor, and $r^{t+1}\left(s_j^{t+1}\right)$ is the immediate reward obtained upon reaching the next-hop state s_j^{t+1}. The term $V_j^t\left(s_j^{t+1}\right)$ is given by

$$V_j^t\left(s_j^{t+1}\right) = \max_{a \in A_j} Q_j^t\left(s_j^{t+1}, a\right). \tag{2}$$

4 System model

4.1 Network model

In this paper, a disaster monitoring scenario based on FANETs is considered. The network consists of n UAVs deployed in a three-dimensional region and one ground control station (GCS) that serves as the data sink, as illustrated in Figure 1. The FANET is modeled as an undirected graph $G = (U, L)$, where $U = \{u_1, u_2, \ldots, u_n, u_{\mathrm{GCS}}\}$ denotes the set of network nodes, including n UAVs and a GCS, and L denotes the set of communication links among these nodes. The main assumptions are summarized as follows.

- The GCS acts as the destination node. Source UAVs collect environmental data and forward it to the GCS through direct transmission or multi-hop relaying. Backbone UAVs (BUs) are one-hop neighbors of the GCS and are responsible for directly forwarding data packets to the GCS.
- Each UAV is assigned a unique identifier to distinguish it from other nodes in the network.
- All UAVs are equipped with a GPS positioning module and an inertial measurement unit (IMU), which provide real-time position and velocity information.
- Each UAV is equipped with a wireless communication module with a maximum communication range r. Two nodes are regarded as neighbors and can establish a communication link if the Euclidean distance between them does not exceed r. In addition, a UAV can directly establish a communication link with the GCS when their distance does not exceed r.
- All UAV nodes adopt the IEEE 802.11n standard at the MAC layer.

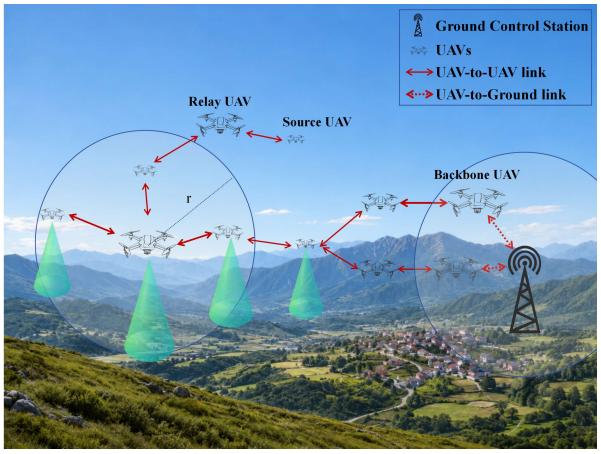


Fig. 1 FANET scenario for disaster monitoring

4.2 Reinforcement Learning Framework for FANETs

In this work, we regard each data packet as an agent and model the FANET routing problem as an MDP. The state space and action space are defined as follows:

State: The state is defined as the UAV node currently holding the data packet. If the packet resides at UAV_i, the corresponding state is denoted as s_i.

Action: The action is defined as forwarding the data packet to one of the current node's neighbors within the single-hop range. When u_i chooses neighbor u_j as the next hop, the corresponding action is denoted as a_j.

To improve the stability of the learning process and accelerate convergence, we adopt a weighted double Q-learning method for FANET routing, which will be detailed in Section 5.2.4. Through continuous interaction with the environment, the Q-values gradually converge, yielding the optimal routing policy.

5 Methods

This section presents the WDQAR routing protocol in detail, including the adaptive neighbor discovery module and the routing decision module.

5.1 Adaptive Neighbor Discovery Module

In FANETs, each UAV periodically broadcasts HELLO messages to its neighboring nodes and receives HELLO messages from them to obtain local topology information. A HELLO message contains the residual energy, the spatial coordinates (x_i^t, y_i^t, z_i^t), and the mobility information $(V_i^t, \theta_i^t, \varphi_i^t)$, where V_i^t is the speed, θ_i^t is the yaw (azimuth) angle, and φ_i^t is the polar angle measured from the z-axis. It also carries the valid neighbor ratio, which reflects the proportion of reliable neighbors around the node and is formally defined in Section 5.2.3. The steps for constructing the neighbor table NT_i of node u_i are as follows:

- When node u_i receives a HELLO message from a neighbor u_j, it first looks up the corresponding entry of u_j in its local neighbor table. If no such entry exists, u_i creates a new one and records the information of u_j carried in the message.
- If u_i subsequently receives another HELLO message from u_j, it updates the state information of u_j in the entry and resets the associated timer.
- If u_i does not receive any HELLO message from u_j within the entry's valid time VT_j, the entry of u_j is removed from the neighbor table.

The high mobility of UAVs causes the network topology to change frequently, which requires the neighbor table to be updated in a timely manner. If the HELLO transmission interval is too long, the information stored in the neighbor table becomes outdated; if it is too short, the control overhead increases excessively. Therefore, an appropriate HELLO interval is essential. To this end, this paper proposes a dynamic adaptive algorithm to adjust the HELLO interval of each UAV. The interval is determined by the link duration and the cosine similarity of velocity directions, and is

bounded within $[HP_{\min}, HP_{\max}]$. At time t, the velocity vector of u_i is given by

$$\boldsymbol{v}_i^t = \big(v_i^x,\, v_i^y,\, v_i^z\big) = V_i^t\big(\sin\varphi_i^t\cos\theta_i^t,\ \sin\varphi_i^t\sin\theta_i^t,\ \cos\varphi_i^t\big), \tag{3}$$

and that of its neighbor u_j is obtained in the same manner. Let $\boldsymbol{p}_i^t = (x_i^t, y_i^t, z_i^t)$ denote the position of u_i at time t. The relative position and relative velocity of u_i with respect to u_j are defined as $\Delta\boldsymbol{p} = \boldsymbol{p}_i^t - \boldsymbol{p}_j^t$ and $\Delta\boldsymbol{v} = \boldsymbol{v}_i^t - \boldsymbol{v}_j^t$, with components $(\Delta x, \Delta y, \Delta z)$ and $(\Delta v_x, \Delta v_y, \Delta v_z)$, respectively.Assuming both UAVs maintain their current velocities,the distance between them after time T is given by

$$d_{ij}^2(T) = \sum_{\xi\in\{x,y,z\}} \big(\Delta\xi + \Delta v_\xi T\big)^2. \tag{4}$$

The predicted link duration LD_{ij} is defined as the time at which $d_{ij}(T)$ first reaches the communication range r. Substituting $d_{ij}(LD_{ij}) = r$ yields the quadratic equation

$$l\,(LD_{ij})^2 + m\,LD_{ij} + n = 0, \tag{5}$$

where $l = \|\Delta\boldsymbol{v}\|^2$, $m = 2\,\Delta\boldsymbol{p}\cdot\Delta\boldsymbol{v}$, and $n = \|\Delta\boldsymbol{p}\|^2 - r^2$. Accordingly, the link duration LD_{ij} is calculated as

$$LD_{ij} = \frac{-m + \sqrt{m^2 - 4ln}}{2l}. \tag{6}$$

where the case $l = 0$ corresponds to two UAVs moving with identical velocities, for which the link is regarded as stable and LD_{ij} is capped at $HP_{\max}$. Furthermore, the cosine similarity between the velocity vectors of u_i and u_j is

$$\mathrm{CS}\big(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t\big) = \frac{\boldsymbol{v}_i^t\cdot\boldsymbol{v}_j^t}{\|\boldsymbol{v}_i^t\|\,\|\boldsymbol{v}_j^t\|}, \tag{7}$$

and its average over the neighbor set N_i^t is

$$\overline{\mathrm{CS}}_i^t = \frac{1}{|N_i^t|}\sum_{j\in N_i^t}\mathrm{CS}\big(\boldsymbol{v}_i^t, \boldsymbol{v}_j^t\big). \tag{8}$$

Finally, the HELLO interval of u_i is set as

$$HP_i^t = \begin{cases} HP_{\min}, & \text{if } \widehat{HP}_i^t < HP_{\min}, \\ HP_{\max}, & \text{if } \widehat{HP}_i^t > HP_{\max}, \\ \widehat{HP}_i^t, & \text{otherwise}, \end{cases} \tag{9}$$

where$\widehat{HP}_i^t = \frac{1+\overline{\mathrm{CS}}_i^t}{2}\cdot\min_{j\in N_i^t} LD_{ij}$ denotes the raw HELLO interval.The pseudocode for the proposed adaptive neighbor discovery process is provided in Algorithm 1.

Algorithm 1 Adaptive Neighbor Discovery

Require:

u_i: i-th node in FANET
NT_i: Neighbor table of u_i
VT_j: Validity period of Hello packets sent by u_j
T_{next}: Next HELLO transmission time of u_i
T_{sim}: Total simulation time

Ensure:

HP_i^t: HELLO interval of u_i at time t
Begin

u_i: Initialize $t = 0$
u_i: $NT_i \leftarrow \emptyset$
u_i: Calculate the initial HELLO interval HP_i^t
u_i: Initialize $T_{\text{next}} = 0$
repeat
    if $t = T_{\text{next}}$ **then**
        u_i: Constructs a HELLO packet containing position, velocity and valid neighbor ratio
        u_i: Broadcast the HELLO packets to neighboring UAVs
        u_i: Calculate HP_i^t according to Equation (9)
        u_i: Update $T_{\text{next}} \leftarrow t + HP_i^t$
    end if
    while u_i receives a hello message from u_j **do**
        u_i: Checks for an entry corresponding to u_j in NT_i
        if NT_i contains an entry corresponding to u_j **then**
            u_i: Update the node information of u_j in this entry
            u_i: Reset VT_j in NT_i
        else
            u_i: Add a new entry corresponding to u_j to NT_i
            u_i: Record the node information of u_j in this entry
            u_i: Set VT_j related to u_j
        end if
    end while
    if an entry for u_j exists in NT_i and no HELLO message is received from u_j until VT_j expires **then**
        u_i: Delete the entry corresponding to u_j from NT_i
    end if
until $t > T_{\text{sim}}$
End

5.2 Routing Decision Module

The routing decision module is responsible for determining the optimal forwarding path for each data packet. Each UAV first filters its candidate neighbor set according to

the geographic location of the destination, and then applies weighted double Q-learning to learn the optimal next-hop decision.

5.2.1 Candidate Neighbor Selection

The forwarding process begins with candidate-neighbor selection. The next hop is not chosen from among all neighbors, but over a filtered candidate set. As shown in Figure 2, sector-based filtering is applied to further narrow the candidate set. A spherical sector is constructed with u_i as the center, the direction from u_i to GCS as the central axis, the communication range r as the radius, and ϕ as the half-cone angle, which is generally set to 60°. The nodes located inside this sector form the primary candidate region S_i, thereby reducing the action space and accelerating the convergence of the routing decision process.

If nodes are available in primary candidate region, the next hop is selected according to the ε-greedy strategy. Otherwise, the node with the minimum angle relative to the target direction is selected from the greedy candidate set C_i as the next hop. The greedy candidate set consists of neighbor nodes closer to GCS than node u_i.

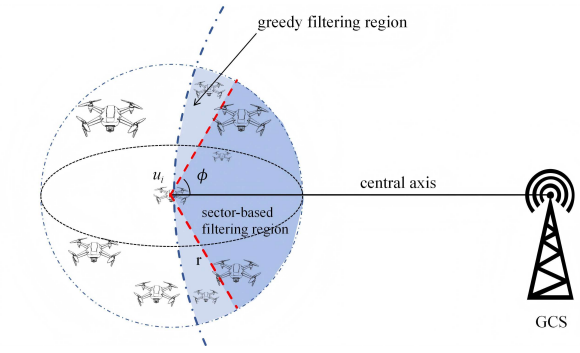


Fig. 2 Candidate neighbor filtering mechanism for next-hop selection

5.2.2 Q-value initialization

In conventional Q-learning, Q-values are typically initialized randomly or to zero. Although simple to implement, such initialization provides no prior estimate of a neighbor's forwarding capability. Without such prior knowledge, the agent must estimate every neighbor's forwarding capability from scratch. This makes early exploration inefficient, slowing convergence and causing performance fluctuations. To address this, we propose a position-based Q-value initialization strategy. It assigns each candidate neighbor an approximate initial Q-value according to its relative position, giving nodes prior knowledge of neighboring states before learning. Specifically, the initial Q-value between forwarding node u_i and candidate neighbor u_j is determined by their relative

distances to the destination GCS, denoted D.

$$Q_{\text{init}}(i,j) = \frac{1}{1 + e^{\frac{d(u_j,D)}{d(u_i,D)}}} \tag{10}$$

where $d(u_j, D)$ and $d(u_i, D)$ denote the distances from candidate neighbor u_j and forwarding node u_i to D, respectively, measured when the first HELLO packet from u_j is received. When u_j is closer to D, the ratio $\frac{d(u_j,D)}{d(u_i,D)}$ decreases, resulting in a larger initial Q-value. Conversely, a larger ratio leads to a smaller initial Q-value.

5.2.3 Reward Function

In this paper, we design the reward function by considering four components: residual energy,link duration,geographic progress and valid neighbor ratio.

- **Residual energy (E_j):** This component represents the communication energy level of u_j. UAV nodes have limited on-board energy for communication, and persistent traffic aggregation on a few nodes rapidly depletes their communication energy, causing link failures and routing disruptions. To alleviate such energy imbalance, we incorporate the residual communication energy into the reward function, steering routing decisions toward nodes with higher remaining energy.
- **Link duration(LD_{ij}):**This component represents the predicted link duration between u_i and u_j.Link duration can accurately characterize the stability and sustainability of communication links in high-mobility FANET scenarios.Selecting nodes with longer expected link durations can avoid link failures and neighbor information expiration, thereby enhancing the overall network stability.
- **Forward progress(F_{ij}):**This component can evaluate the forward progress capability of the next-hop node. Selecting a next-hop node that is closer to the destination and has a smaller forwarding deviation angle λ enables the data packet to advance effectively toward the destination at each hop, reducing the number of forwarding hops and transmission delay. As shown in Figure 3, if two nodes u_j and u_k are equidistant from the destination, the node u_j with a smaller forwarding deviation angle λ_{ij} is closer to the sending node u_i, resulting in a shorter and more stable link.The F_{ij} is given as follows:

$$F_{ij} = \left(1 - \frac{d_{jD}}{d_{iD}}\right) \exp\left(-\frac{\lambda_{ij}^2}{2\left(\pi/4\right)^2}\right), \quad 0 \le \lambda_{ij} \le \frac{\pi}{2} \tag{11}$$

- **Valid neighbor ratio(VNR_{ij}):**This component reflects the effective forwarding potential of the next-hop u_j.A higher ratio indicates that the node has more available neighbors toward the destination. These neighbors provide more forwarding opportunities in subsequent hops. As a result, the probability of routing holes can

be reduced. The VNR_{ij} is given as follows:

$$VNR_{ij} = \begin{cases} 1, & \text{if } u_j \text{ is a BU} \\ \dfrac{|C_j|}{|N_j| + \varepsilon}, & \text{otherwise} \end{cases} \tag{12}$$

where $|C_j|$ and $|N_j|$ represent the number of neighbors in the candidate region of u_j and the total number of neighbors, respectively, and ε is a small positive constant to avoid division by zero.If u_j is a BU, it can deliver packets directly to the destination without relying on any subsequent relay; hence its valid neighbor ratio is set to the maximum value of 1.

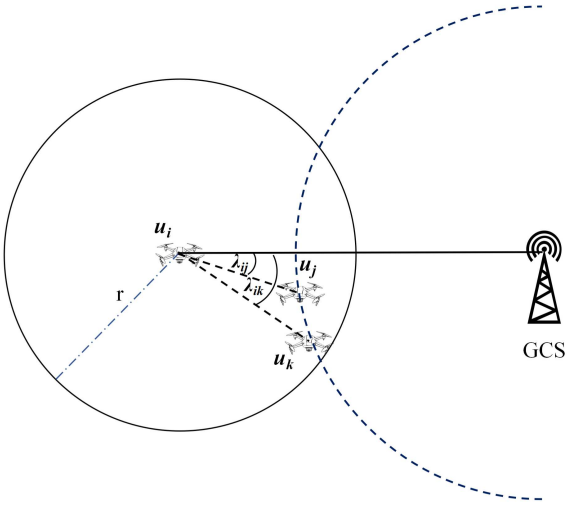


Fig. 3 Illustration of the forward progress

Each candidate next-hop node u_j has four reward components. The reward matrix $\mathbf{X}^t = [\mathbf{x}_j]_{j=1}^{|C_i|}$ is constructed by collecting the reward component vectors of all candidate nodes in C_i, where the reward component vector of u_j is defined as

$$\mathbf{x}_j = [x_{j1}, x_{j2}, x_{j3}, x_{j4}] = [E_j, LD_{ij}, F_{ij}, VNR_{ij}] \tag{13}$$

First, the reward matrix is normalized as

$$x_{jk}^{norm} = \frac{x_{jk} - \min\{x_k\}}{\max\{x_k\} - \min\{x_k\}} \tag{14}$$

where x_{jk} denotes the k-th reward component of candidate u_j ($1 \leq j \leq |C_i|$, $1 \leq k \leq 4$). The information entropy of the k-th reward component is defined as

$$H_k = -\frac{1}{\ln |C_i|} \sum_{j=1}^{|C_i|} p_{jk} \ln p_{jk}. \tag{15}$$

where $p_{jk} = \frac{x_{jk}^{norm}}{\sum_{j=1}^{|C_i|} x_{jk}^{norm}}$ represents the proportion of the k-th reward component of u_j. The entropy weight of the k-th reward component is defined as

$$\omega_k = \frac{1 - H_k}{\sum_{i=1}^{4}(1 - H_i)}. \tag{16}$$

Therefore, the immediate reward r_i^t is defined as follows:

$$r_i^t(j) = \begin{cases} r_{\max} & \text{if } u_j \text{ is destination} \\ r_{\min} & \text{if ACK timeout or routing void} \\ \sum_{k=1}^{4} \omega_k x_{jk}^{norm} & \text{otherwise} \end{cases} \tag{17}$$

If u_j is the destination node, the reward is set to the maximum value, indicating that the data packet is successfully delivered to the destination. If no ACK packet is received within the timeout period or a routing void occurs, the reward is set to the minimum value. Otherwise, the immediate reward is calculated as the weighted sum of the four components.

5.2.4 Weighted double Q-learning update

Conventional Q-learning uses the maximum Q-value to estimate the future return, where the same Q-table is used for both action selection and value evaluation. This may introduce overestimation bias, especially in stochastic and highly dynamic environments. Double Q-learning alleviates this problem by maintaining two independent Q-tables, where one is used to select actions and the other to evaluate their values [29]. However, the double estimator may sometimes underestimate action values. To balance the overestimation of Q-learning and the underestimation of double Q-learning, this paper adopts weighted double Q-learning, which combines the two Q-values through an adaptive weight to obtain a more accurate action-value estimate and improve the convergence stability of learning [30].

In WDQAR, each UAV maintains two independent Q-tables, denoted as Q_1 and Q_2. For a forwarding node u_i, the state is denoted as s_i, and the action a_j represents selecting candidate neighbor u_j as the next hop. During each Q-value update, one of the two Q-tables is randomly selected for updating, while the other is used to construct the weighted future value.If Q_1 is selected for update, the best and worst actions in the next state s' are first determined according to Q_1:

$$a_1^* = \arg \max_{a \in A(s')} Q_1(s', a) \tag{18}$$

$$a_1^L = \arg \min_{a \in A(s')} Q_1(s', a). \tag{19}$$

The adaptive weight is then computed from the spread of the other estimator Q_2 between these two actions:

$$\beta_1 = \frac{\left|Q_2(s', a_1^*) - Q_2(s', a_1^L)\right|}{c + \left|Q_2(s', a_1^*) - Q_2(s', a_1^L)\right|}, \tag{20}$$

where $c > 0$ controls the balance between the single and double estimators. The weighted future value combines the two estimators at the maximizing action a_1^*:

$$Q_1^W(s') = \beta_1 Q_1(s', a_1^*) + (1 - \beta_1) Q_2(s', a_1^*). \tag{21}$$

Accordingly, Q_1 is updated as

$$Q_1(s_i, a_j) \leftarrow Q_1(s_i, a_j) + \alpha \left[r + \gamma Q_1^W(s') - Q_1(s_i, a_j)\right], \tag{22}$$

where α is the learning rate, γ is the discount factor, and r is the immediate reward. If Q_2 is selected instead, the update is symmetric. Through this weighted mechanism, WDQAR balances the overestimation of conventional Q-learning and the underestimation of double Q-learning, thereby improving the stability of routing value estimation in dynamic FANET environments.

5.2.5 Dynamic learning factors

In order to quickly respond to topological changes and accelerate convergence, this paper adaptively adjusts the learning parameters in Q-learning, including learning rate and discount factor.

- **Learning rate (α_{ij}):** The learning rate controls the update magnitude of the value function and determines the learning speed of the Q-learning algorithm by balancing new and previous information. Its value is bounded within $[0, 1]$. In WDQAR, the learning rate is dynamically adjusted according to the relative delay of the selected next-hop. When the next-hop delay is larger than the average of the candidate neighbors, the link is unstable, so a larger learning rate is adopted to update the Q-value more rapidly. Conversely, when the delay is smaller than the average, the link is more reliable, and a smaller learning rate is adopted so that decisions rely more on historical experience [31]. The delay of the candidate neighbor u_j is standardized as

$$delay_{ij}^{standard} = \frac{delay_{ij} - \mu_{delay}^i}{\sigma_{delay}^i} \tag{23}$$

  where μ_{delay}^i and σ_{delay}^i are the mean and standard deviation of the delays between u_i and its candidate neighbors, respectively, computed as

$$\mu_{delay}^i = \frac{1}{|C_i|} \sum_{j=1}^{|C_i|} delay_{ij}, \tag{24}$$

$$\sigma^{i}_{delay} = \sqrt{\frac{1}{|C_i|}\sum_{j=1}^{|C_i|}\left(delay_{ij} - \mu^{i}_{delay}\right)^2}. \tag{25}$$

The learning rate α_{ij} is then calculated as

$$\alpha_{ij} = \begin{cases} 0.1, & \text{if } delay_{ij}^{standard} \leq 0 \\ 1 - e^{-delay_{ij}^{standard}}, & \text{if } delay_{ij}^{standard} > 0 \end{cases} \tag{26}$$

- **Discount factor (γ_{ij}):** The discount factor $\gamma_{ij} \in [0, 1]$ reflects the importance that the agent attaches to future rewards. The more drastic the topology variation, the more uncertain the future path state becomes; over-relying on future rewards under such conditions would bias the routing decision. Therefore, γ_{ij} is formulated as

$$\gamma_{ij} = \sqrt{\frac{\left|N_i^t \cap N_i^{t-1}\right|}{\left|N_i^t \cup N_i^{t-1}\right|} \cdot \frac{\left|N_j^t \cap N_j^{t-1}\right|}{\left|N_j^t \cup N_j^{t-1}\right|}}, \tag{27}$$

  where N_i^t, N_i^{t-1} and N_j^t, N_j^{t-1} denote the neighbor sets of u_i and u_j in two consecutive periods t and $t-1$, respectively.

5.2.6 Routing forwarding procedure

When a source node generates a data packet, it first determines whether the destination GCS lies within its one-hop communication range. If the GCS is a direct neighbor, the packet is delivered to it directly in a single hop; otherwise, the node selects from its neighbors those UAVs that are geographically closer to the GCS to form a candidate neighbor set, and then applies an ϵ-greedy policy over this set to choose the next hop for forwarding, while simultaneously starting an ACK timeout timer. Upon receiving the data packet, the relay node immediately returns an ACK to its previous hop and continues forwarding the packet to the next hop; based on the received ACK, the previous-hop node computes a reward and updates the corresponding Q-value, so that the forwarding policy is continuously optimized through ongoing interaction. Through multi-hop forwarding by relay UAVs, the data packet is progressively relayed to a backbone UAV, which finally delivers it to the GCS in a single hop. Algorithm 2. describes the pseudo-code related to the routing forwarding procedure.

6 Results

In this section, we use Network Simulator 3 (NS-3) [32] to evaluate the performance of the proposed WDQAR routing protocol.First, weighted double Q-learning is compared with single Q-learning to assess the effect of the weighted double Q-learning on the convergence behavior of Q-values. The comparison focuses on two aspects: the volatility and stability of the Q-values. Then, WDQAR is compared with QFANET[24] and QGEO[22] to evaluate its performance under different network scenarios. The main evaluation scales include packet delivery ratio, average end-to-end delay, residual energy, and routing overhead.

Algorithm 2 Routing Forwarding Procedure

Require:

u_i: The i-th node in FANET

ϵ, α, γ: Learning parameters

u_{GCS}: Ground control station

C_i: Candidate neighbor set of u_i

Ensure:

Q-table stored in u_i

Begin

while a data packet p is generated or received by u_i **do**
  if u_i is a BU **then**
    u_i: deliver packet p to u_{GCS} in a single hop
  else
    u_i: compute the candidate set C_i according to Section 5.2.1
    for each $u_j \in C_i$ **do**
      if the Q-value of u_j has not been visited **then**
        u_i: initialize its Q-value according to the relative position
      end if
    end for
    u_i: generate a random value $v_{\text{rand}} \in [0, 1]$
    if $v_{\text{rand}} < \varepsilon$ **then**
      u_i: randomly select u_j from C_i
    else
      u_i: select the greedy u_j according to the Q-table
    end if
    u_i: forward the data packet to the selected u_j
    u_i: wait for an ACK from u_j
    if u_j is trapped into a routing hole or ACK timeout **then**
      $r_i^t(j) = r_{\min}$
    else if $u_j = u_{GCS}$ **then**
      $r_i^t(j) = r_{\max}$
    else
      $r_i^t(j) = \sum_{k=1}^{4} \omega_k x_{jk}^{\text{norm}}$
    end if
    u_i: update the Q-table based on $r_i^t(j)$
  end if
end while

End

6.1 Simulation settings

In our simulation scenario, the FANET consists of 40 to 100 UAVs randomly distributed in a three-dimensional space of $1000 \times 1000 \times 500\,\text{m}^3$. All UAVs fly at a speed of $10 - 30\,\text{m/s}$ and follow the three-dimensional Gauss-Markov (3D GM) mobility model. We randomly select 5 UAV nodes as source nodes, set the destination node as

a ground control station, and the remaining nodes serve as relay nodes. The detailed simulation parameters are summarized in Table 2. All experiments are repeated 10 times independently, and the final results are presented as the average value. The

Table 2 Simulation parameters.

Parameter	Value
Simulation tool	NS3
Network dimensions	$1000 \times 1000 \times 500$ m^3
Number of UAVs in the network	40–100 nodes
Number of source UAVs	5
Velocity of UAVs	10–30 m/s
Communication radius	300 m
Energy consumption model	First Order Radio Model
Initial communication energy	200 J
Movement model	3D Gauss–Markov
Communication standard	IEEE 802.11n
Traffic model	Constant bit rate (CBR)
Data rate	2 Mbps
Packet size	512 bytes
Transport protocol	UDP

definitions of each evaluation scale are given as follows:

- **Packet delivery ratio(PDR):**This evaluation scale is defined as the ratio of the total number of data packets successfully received by the destination node to the total number of data packets sent by all source nodes.

$$\mathrm{PDR} = \frac{PK_{\mathrm{recv}}}{\sum_{i=1}^{n_s} PK_{\mathrm{send}}^{i}} \times 100\% \tag{28}$$

  where PK_{recv}, PK_{send}^{i} and n_s denote the total number of data packets received by the destination node, the number of data packets transmitted by the i-th source node, and the total number of source nodes, respectively.
- **End-to-end delay(EED):**This evaluation scale is defined as the average time required for data packets to be transmitted from the source node to the destination node.

$$\mathrm{EED} = \frac{1}{PK_{\mathrm{recv}}} \sum_{i=1}^{PK_{\mathrm{recv}}} (t_{\mathrm{recv},i} - t_{\mathrm{send},i}) \tag{29}$$

  where PK_{recv} denotes the total number of successfully received packets at the destination node, $t_{\mathrm{recv},i}$ denotes the reception time of the i-th packet, and $t_{\mathrm{send},i}$ denotes the transmission time of the corresponding packet at the source node.

- **Consumed communication energy ($\bar{E}_{\mathrm{c}}$):** This evaluation scale is defined as the average consumed energy of all nodes in the network.

$$\bar{E}_{\mathrm{c}} = \frac{1}{N} \sum_{i=1}^{N} E_{\mathrm{c},i} \tag{30}$$

  where N denotes the total number of UAV nodes in the network, and $E_{\mathrm{c},i}$ denotes the total consumed communication energy of the i-th UAV node.
- **Routing overhead(RO):**This evaluation scale represents the ratio of the number of control packets to the number of transmitted data packets in the data transmission process.

$$RO = \frac{N_{\mathrm{HELLO}}}{N_{\mathrm{data}}} \times 100\%, \tag{31}$$

  where N_{HELLO} denotes the total number of transmitted HELLO packets, and N_{data} denotes the total number of transmitted data packets.

6.2 Q-Value convergence analysis

In RL-based routing, the Q-value of a candidate next-hop represents the expected return of forwarding to it under the current state. It also indirectly reflects the forwarding advantage of next hop nodes, such as a stable communication link and sufficient residual energy. In this paper, volatility and stability are adopted to characterize the variation of Q-values. Specifically, volatility is measured by the variance of Q-value changes, while stability is measured by the cumulative sum of the absolute differences between Q-values at consecutive time steps.

As a comparison method, single Q-learning uses a single Q-table for both action selection and value evaluation. As shown in Figure 4, as the number of UAV nodes increases from 40 to 100, the Q-value volatility of weighted double Q-learning is consistently lower than that of single Q-learning, with an average reduction of approximately 45.62%. Meanwhile, the total Q-value variation of weighted double Q-learning is also consistently smaller than that of single Q-learning, with an average reduction of approximately 33.03%, as shown in Figure 5. These results indicate that weighted double Q-learning can effectively suppress Q-value estimation fluctuations and make the learning process more stable. This is because it separates action selection from value evaluation, thereby reducing the estimation bias caused by the maximization operation in single Q-table updates and further decreasing frequent routing strategy switching among candidate nodes.

6.3 Packet delivery ratio

Figure 6 shows the variation of packet delivery ratio with the number of UAV nodes for different routing protocols. The PDR of all routing protocols increases with the number of UAV nodes. This is because a denser network provides more available next-hop nodes, reducing route failures caused by node mobility and link disconnections. Consequently, more packets are successfully delivered to their destinations, leading to a higher PDR. Compared with QFANET and QGEO, WDQAR achieves

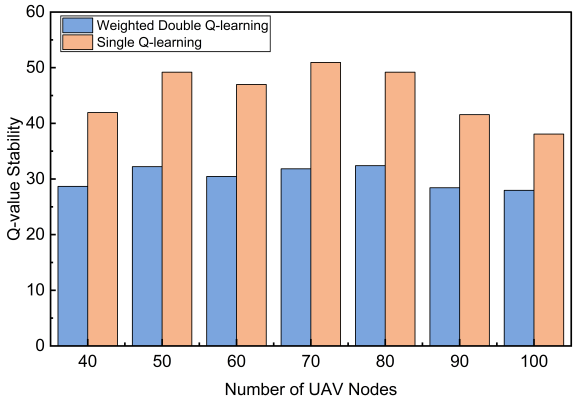


Fig. 4 Q-value volatility for varying node densities

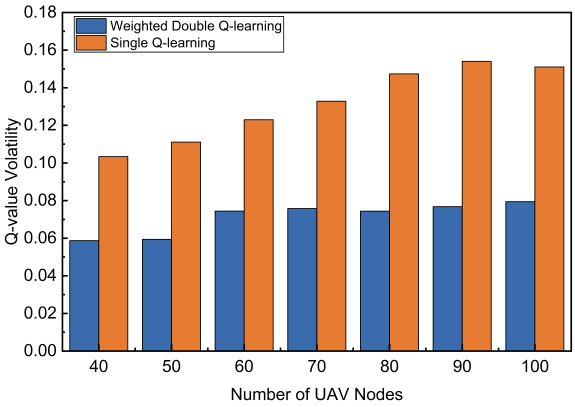


Fig. 5 Q-value stability for varying node densities

the highest PDR under all network scales. Specifically, the average PDR of WDQAR reaches 92.62%. WDQAR improves the PDR by approximately 3.42% compared with QFANET and 6.50% compared with QGEO.

Figure 7 shows the variation of packet delivery ratio with UAV speed for different routing protocols. The PDR of all routing protocols gradually decreases as the UAV speed increases.This is because higher UAV speeds lead to more frequent topology changes and shorter link lifetimes, which increase the probability of link disconnections during packet forwarding and cause more route failures and packet losses. Compared with QFANET and QGEO, WDQAR achieves the highest PDR across all UAV speeds. Specifically, the average PDR of WDQAR reaches 93.78%, approximately 2.40 and 4.77 percentage points higher than those of QFANET and QGEO, respectively.

Overall, Figure 6 and Figure 7 show that WDQAR achieves a higher PDR under different node densities and UAV speeds. This improvement is mainly attributed to its reward function, which prioritizes next-hop UAVs with longer link duration and

a higher proportion of valid neighbors, thereby reducing link disconnections, routing holes, and packet losses. In addition, the weighted double Q-learning mechanism suppresses overestimation bias and provides more accurate Q-value estimates, enabling WDQAR to identify optimal routes more precisely and further improve the PDR.

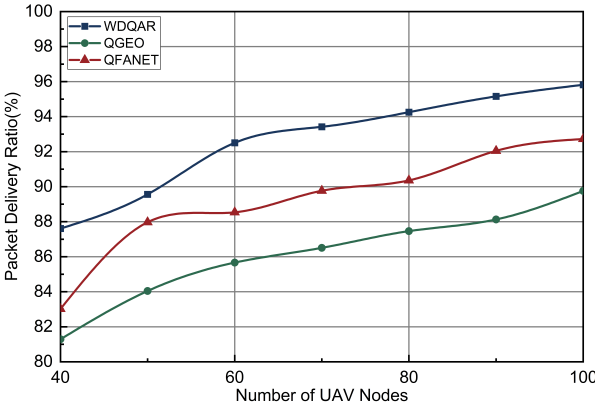


Fig. 6 Packet delivery rate for varying node densities

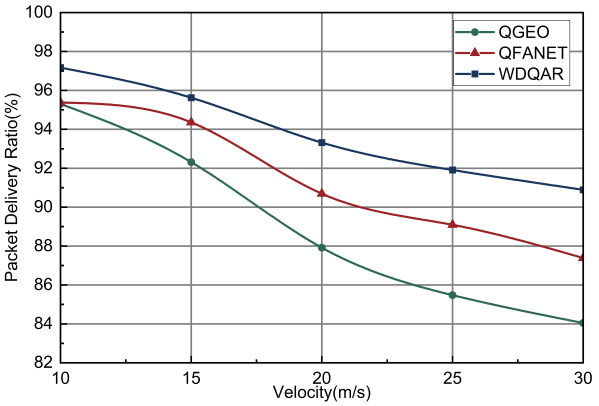


Fig. 7 Packet delivery rate for varying UAV velocities

6.4 End-to-end delay

Figure 8 shows the average end-to-end delay versus the number of UAV nodes for different routing protocols. As the number of UAV nodes increases, the delay of all protocols rises, mainly because a larger network leads to more hops from the source to the destination, and each additional hop introduces extra transmission and queuing delay. Compared with QFANET and QGEO, WDQAR achieves the lowest delay under

all network scales. Specifically, its average delay is 16.80 ms, which is approximately 40.93% and 44.30% lower than those of QFANET and QGEO, respectively.

Figure 9 shows the variation of average end-to-end delay with UAV speed for different routing protocols. The end-to-end delay of all routing protocols increases with UAV speed, because rapid topology changes cause frequent link breakages, which in turn trigger packet retransmissions. WDQAR maintains the lowest delay across all UAV speeds, with an average delay of 18.78 ms. Compared with QFANET and QGEO, WDQAR reduces the average delay by approximately 35.82% and 39.31%, respectively.

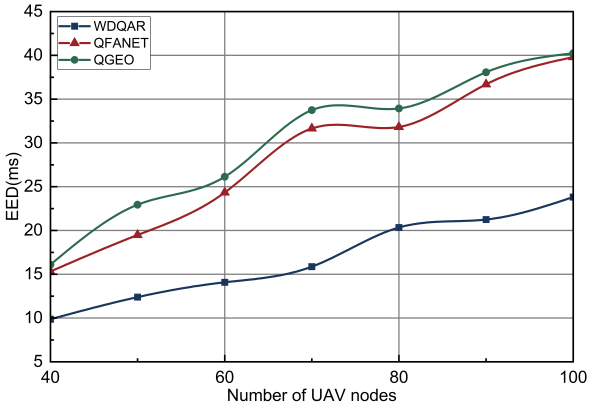


Fig. 8 End-to-end delay for varying node densities

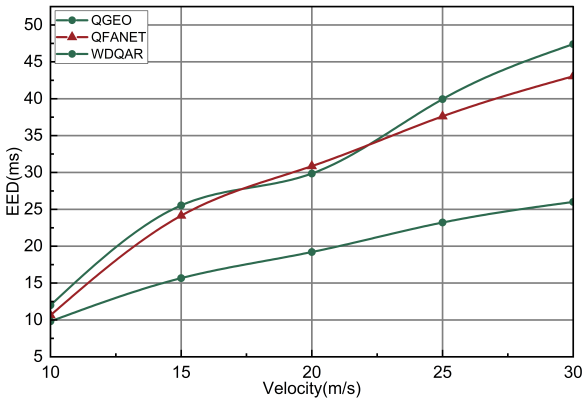


Fig. 9 End-to-end delay for varying UAV velocities

WDQAR achieves the lowest delay mainly for three reasons. First, its candidate neighbor filtering restricts next-hop selection to a smaller set of valid neighbors, which

shrinks the action space and accelerates convergence. Second, the geographic progress metric in the reward function favors next hops closer to the destination, reducing unnecessary hops and detours. In addition, the distance-based Q-value initialization accelerates the convergence to the optimal route, and thus reduces the end-to-end delay.

6.5 Consumed communication energy

Figure 10 shows the variation of average communication energy consumption with the number of UAV nodes for different routing protocols. A larger number of UAV nodes leads to higher energy consumption for all routing protocols. This is because more UAV nodes result in more neighbors and control-message exchanges, increasing the energy consumed in packet transmission and reception. WDQAR achieves the lowest energy consumption under all network scales, with an average value of 11.37 J. Compared with QFANET and QGEO, WDQAR reduces the average energy consumption by approximately 12.34%, and 6.86%, respectively.

Figure 11 shows the variation of average energy consumption with UAV speed for different routing protocols.The energy consumption increases with UAV speed because faster movement shortens the link duration and causes more frequent topology changes, which require additional HELLO message exchanges and packet retransmissions.WDQAR maintains the lowest energy consumption across all UAV speeds, with an average value of 11.40 J. Compared with QFANET and QGEO, WDQAR reduces the average energy consumption by approximately 9.23% and 4.21%, respectively.

Overall, Figure 10 and Figure 11 demonstrate the energy efficiency of WDQAR. Its adaptive HELLO mechanism reduces unnecessary control-message transmissions by adjusting the HELLO interval according to network mobility. Meanwhile, the residual-energy term in the reward function encourages the selection of energy-sufficient next-hop UAVs, thereby balancing forwarding loads and reducing energy consumption.

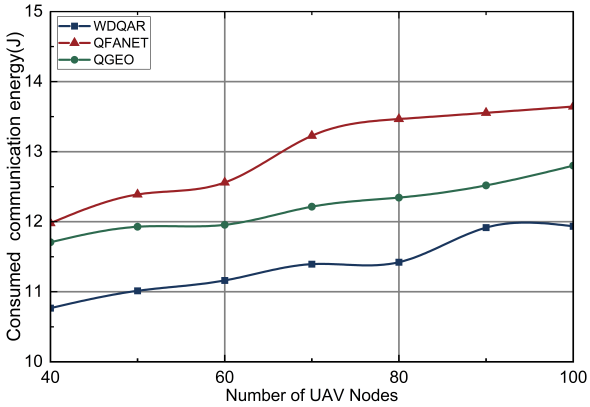


Fig. 10 Consumed communication energy for varying node densities

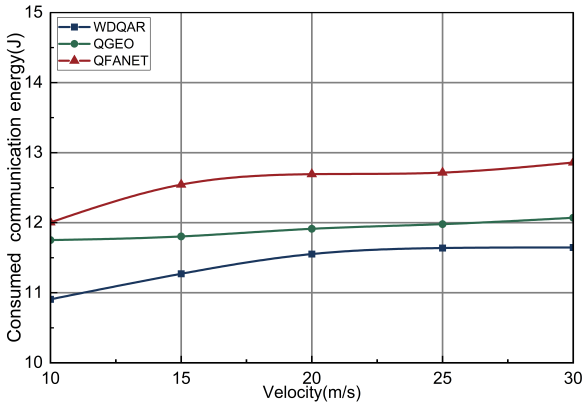


Fig. 11 Consumed communication energy for varying UAV velocities

6.6 Routing overhead

Figure 12 shows the variation of routing overhead with the number of UAV nodes.The routing overhead increases with network size because a larger number of UAVs results in more HELLO packet transmissions and receptions. WDQAR achieves the lowest routing overhead under all network scales, with an average value of 9.94%. Compared with QFANET and QGEO, WDQAR reduces the average routing overhead by approximately 29.43% and 34.80%, respectively.

Figure 13 shows the variation of routing overhead with UAV speed. The routing overhead of WDQAR increases with UAV speed because faster mobility shortens the link duration and reduces mobility similarity, causing the adaptive HELLO mechanism to exchange neighbor information more frequently. In contrast, the overheads of QFANET and QGEO fluctuate only slightly because their HELLO intervals are fixed. WDQAR maintains the lowest average routing overhead of 11.67%, which is approximately 31.88% and 37.19% lower than those of QFANET and QGEO, respectively.

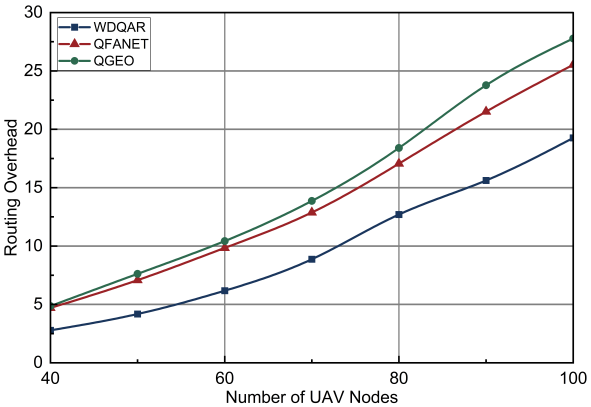


Fig. 12 Routing overhead for varying node densities

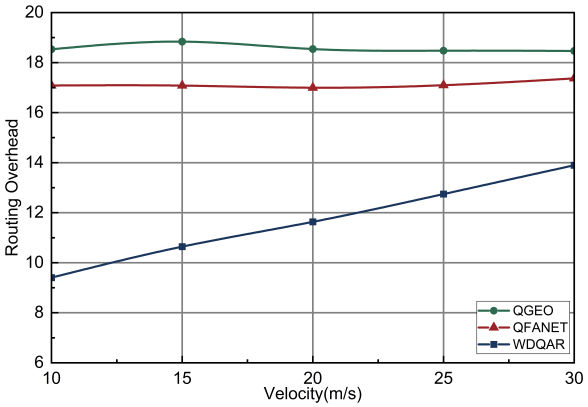


Fig. 13 Routing overhead for varying UAV velocities

Overall, WDQAR reduces unnecessary control-message transmissions by using longer HELLO intervals when the topology is stable, while shortening the intervals when links become unstable to maintain timely neighbor information.

7 Conclusion

This paper proposed WDQAR, an adaptive routing protocol based on weighted Double Q-learning for highly dynamic FANETs. WDQAR adaptively adjusts the HELLO interval to reduce control overhead, filters candidate next hops by geographic progress to shrink the action space, and weights the future-value estimates of two Q-tables to mitigate overestimation bias and improve the accuracy of Q-value estimates. In this protocol, a reward function is designed based on residual energy, link duration, geographic progress, and valid-neighbor ratio. In addition, a Q-value initialization strategy and adaptive learning parameters are introduced to further accelerate the learning process. The simulation results show that WDQAR achieves more stable Q-value updates than single Q-learning. It also provides a higher packet delivery ratio and lower end-to-end delay than QFANET and QGEO under different network densities and UAV speeds. Overall, WDQAR improves routing reliability and learning stability in highly dynamic FANETs.

Future work will extend WDQAR to larger and more complex FANET scenarios and further improve its adaptability to dynamic network conditions. In addition, hardware-in-the-loop and real-world UAV experiments will be conducted to examine the practical feasibility of the proposed protocol.

Declarations

- Funding:Not applicable
- Competing interests:The author(s) declare no competing interests.
- Ethics approval and consent to participate:Not applicable
- Consent for publication:Not applicable

- Data availability :The datasets used and analysed during the current study available from the corresponding author on reasonable request.
- Materials availability:Not applicable
- Code availability :Some or all models, or code generated or used during the study are available from the corresponding author by request.
- Author contribution: Qingzhong Ni: Conceptualization, Methodology, Software, Formal analysis, Data curation, Visualization, Writing -original draft. Yijie Bai: Investigation, Validation, Data curation, Visualization. Daojie Yu: Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Writing-review & editing. Qi Zhang: Formal analysis, Validation, Investigation. Fan Meng: Resources, Writing-review & editing, Validation.

References

[1] Singh, P.K., Sharma, A.: An intelligent WSN-UAV-based IoT framework for precision agriculture application. Comput. Electr. Eng. **100**, 107912 (2022) https://doi.org/10.1016/j.compeleceng.2022.107912

[2] Jiao, Z.: The application of remote sensing techniques in ecological environment monitoring. Highl. Sci. Eng. Technol. **81**, 449–455 (2024) https://doi.org/10.54097/7dqegz64

[3] Sun, G., He, L., Sun, Z., Wu, Q., Liang, S., Li, J., Niyato, D., Leung, V.C.: Joint task offloading and resource allocation in aerial-terrestrial UAV networks with edge and fog computing for post-disaster rescue. IEEE Trans. Mob. Comput. **23**(9), 8582–8600 (2024) https://doi.org/10.1109/TMC.2024.3350886

[4] Pei, Y., Sun, H., Sun, Q.: Optimizing UAV-UGV system for reconnaissance and strike based on soft actor critic algorithm. In: 2025 IEEE 14th Data Driv. Control Learn. Syst., pp. 899–903 (2025). https://doi.org/10.1109/DDCLS66240.2025.11066058.

[5] Chriki, A., Touati, H., Snoussi, H., Kamoun, F.: FANET: Communication, mobility models and security issues. Comput. Netw. **163**, 106877 (2019) https://doi.org/10.1016/j.comnet.2019.106877

[6] Yin, H., Wei, J., Zhao, H., Zhang, J., Wang, H., Ren, B.: Intelligent communication and networking key technologies for manned/unmanned cooperation: States-of-the-art and trends. J. Commun. **45**, 1–17 (2024) https://doi.org/10.11959/j.issn.1000-436x.2024037

[7] Du, X., Cao, Y., Wang, D., Lv, C., Wu, C., Wang, K.: A tri-phases message oriented trust model in FANET. IEEE Trans. Netw. Sci. Eng. **11**(6), 5298–5310 (2024) https://doi.org/10.1109/TNSE.2024.3374733.

[8] Srivastava, A., Prakash, J.: Future FANET with application and enabling techniques: Anatomization and sustainability issues. Comput. Sci. Rev. **39**, 100359

(2021) https://doi.org/10.1016/j.cosrev.2020.100359

[9] Fatemidokht, H., Rafsanjani, M.K., Arya, V., Gupta, B.B.: Routing protocols in FANETs: A comprehensive survey and taxonomy. J. Syst. Archit., 103774 (2026) https://doi.org/10.1016/j.sysarc.2026.103774

[10] Almansor, M.J., Din, N.M., Baharuddin, M.Z., Ma, M., Alsayednoor, H.M., Al-Shareeda, M.A., Al-asadi, A.J.: Routing protocols strategies for flying Ad-Hoc network (FANET): Review, taxonomy, and open research issues. Alex. Eng. J. **109**, 553–577 (2024) https://doi.org/10.1016/j.aej.2024.09.032

[11] Rahmani, A.M., Hussain, D., Ismail, R.J., Alanazi, F., Belhaj, S., Yousefpoor, M.S., Yousefpoor, E., Darwesh, A., Hosseinzadeh, M.: An adaptive and multi-path greedy perimeter stateless routing protocol in flying ad hoc networks. Veh. Commun. **50**, 100838 (2024) https://doi.org/10.1016/j.vehcom.2024.100838

[12] Gangopadhyay, S., Jain, V.K.: A position-based modified OLSR routing protocol for flying ad hoc networks. IEEE Trans. Veh. Technol. **72**(9), 12087–12098 (2023) https://doi.org/10.1109/TVT.2023.3265704

[13] Yang, S., Zhao, W., Wang, C.-M., Dong, W.-Y., Ju, X.: Betweenness centrality based dynamic source routing for flying ad hoc networks in marching formation. IEEE Trans. Veh. Technol. **74**(8), 12791–12798 (2025) https://doi.org/10.1109/TVT.2025.3550336

[14] Mansour, H.K., Mutar, M.H., Aziz, I.A., Mostafa, S.A., Mahdin, H., Abbas, A.H., Hassan, M.H., Abdulsattar, N.F., Jubair, M.A.: Cross-layer and energy-aware AODV routing protocol for flying ad-hoc networks. Sustainability **14**(15), 8980 (2022) https://doi.org/10.3390/su14158980

[15] Lakew, D.S., Sa'ad, U., Dao, N.-N., Na, W., Cho, S.: Routing in flying ad hoc networks: A comprehensive survey. IEEE Commun. Surv. Tutor. **22**(2), 1071–1120 (2020) https://doi.org/10.1109/COMST.2020.2982452

[16] Tho, M.C., Vo, T.T., *et al.*: GPSR-CB: A novel routing algorithm for FANET using cross-layer models in combination with multi-level backbone UAV. Ad Hoc Netw. **173**, 103828 (2025) https://doi.org/10.1016/j.adhoc.2025.103828

[17] Cui, J., Ma, L., Wang, R., Liu, M.: Research and optimization of GPSR routing protocol for vehicular ad-hoc network. China Commun. **19**(10), 194–206 (2022) https://doi.org/10.23919/JCC.2022.00.031

[18] Hosseinzadeh, M., Tanveer, J., Ionescu-Feleaga, L., Ionescu, B.-S., Yousefpoor, M.S., Yousefpoor, E., Ahmed, O.H., Rahmani, A.M., Mehmood, A.: A greedy perimeter stateless routing method based on a position prediction mechanism for flying ad hoc networks. J. King Saud Univ. – Comput. Inf. Sci. **35**(8), 101712 (2023) https://doi.org/10.1016/j.jksuci.2023.101712

[19] Yang, Z., Xiong, Y., Shangguan, Y., Wang, Z., Yin, Y.: A GPSR protocol based on mobile prediction and node stability. In: Proc. 2024 Int. Conf. Power Electron. Artif. Intell., pp. 449–456 (2024). https://doi.org/10.1145/3674225.3674306

[20] Bezziane, M.B., Sahraoui, Y., Brik, B., Mekkas, L., Bougeurra, S., Khaldi, A.: On the performance evaluation of mobility model-based GPSR routing protocol in flying ad hoc networks. In: 2024 6th Int. Conf. Pattern Anal. Intell. Syst., pp. 1–7 (2024). https://doi.org/10.1109/PAIS62114.2024.10541298.

[21] Watkins, C.J., Dayan, P.: Q-learning. Mach. Learn. **8**(3), 279–292 (1992) https://doi.org/10.1007/BF00992698

[22] Jung, W.-S., Yim, J., Ko, Y.-B.: QGeo: Q-learning-based geographic ad hoc routing protocol for unmanned robotic networks. IEEE Commun. Lett. **21**(10), 2258–2261 (2017) https://doi.org/10.1109/LCOMM.2017.2656879

[23] Liu, J., Wang, Q., He, C., Jaffrès-Runser, K., Xu, Y., Li, Z., Xu, Y.: QMR: Q-learning based multi-objective optimization routing protocol for flying ad hoc networks. Comput. Commun. **150**, 304–316 (2020) https://doi.org/10.1016/j.comcom.2019.11.011

[24] Costa, L.A.L.F., Kunst, R., Freitas, E.P.: Q-FANET: Improved q-learning based routing protocol for FANETs. Comput. Netw. **198**, 108379 (2021) https://doi.org/10.1016/j.comnet.2021.108379

[25] Arafat, M.Y., Moh, S.: A q-learning-based topology-aware routing protocol for flying ad hoc networks. IEEE Internet Things J. **9**(3), 1985–2000 (2021) https://doi.org/10.1109/JIOT.2021.3089759

[26] Cui, Y., Zhang, Q., Feng, Z., Wei, Z., Shi, C., Yang, H.: Topology-aware resilient routing protocol for FANETs: An adaptive q-learning approach. IEEE Internet Things J. **9**(19), 18632–18649 (2022) https://doi.org/10.1109/JIOT.2022.3162849

[27] Hosseinzadeh, M., Ali, S., Ionescu-Feleaga, L., Ionescu, B.-S., Yousefpoor, M.S., Yousefpoor, E., Ahmed, O.H., Rahmani, A.M., Mehmood, A.: A novel q-learning-based routing scheme using an intelligent filtering algorithm for flying ad hoc networks (FANETs). J. King Saud Univ. – Comput. Inf. Sci. **35**(10), 101817 (2023) https://doi.org/10.1016/j.jksuci.2023.101817

[28] Rahmani, A.M., Haider, A., Zaidi, M.M., Alanazi, A., Alsubai, S., Alqahtani, A., Yousefpoor, M.S., Yousefpoor, E., Hosseinzadeh, M.: QRCF: A new q-learning-based routing approach using a smart cylindrical filtering system in flying ad hoc networks. Veh. Commun. **53**, 100905 (2025) https://doi.org/10.1016/j.vehcom.2025.100905

[29] Hasselt, H.: Double q-learning. Adv. Neural Inf. Process. Syst. **23** (2010)

[30] Zhang, Z., Pan, Z., Kochenderfer, M.J.: Weighted double q-learning. In: Proc. 26th Int. Jt. Conf. Artif. Intell., pp. 3455–3461 (2017). https://doi.org/10.24963/ijcai.2017/483

[31] Alam, M.M., Moh, S.: Joint topology control and routing in a UAV swarm for crowd surveillance. J. Netw. Comput. Appl. **204**, 103427 (2022) https://doi.org/10.1016/j.jnca.2022.103427

[32] Riley, G.F., Henderson, T.R.: The ns-3 network simulator. In: Wehrle, K., Güneş, M., Gross, J. (eds.) Modeling and Tools for Network Simulation, pp. 15–34. Springer, Berlin, Heidelberg (2010)

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- DataAvailability.pdf