A Lightweight Reinforcement-Learning-Based Real-Time Path-Planning Method for Unmanned Aerial Vehicles

Meng Xi, *Member, IEEE*, Huiao Dai, Jingyi He, Wenjie Li, Jiabao Wen, *Member, IEEE*, Shuai Xiao, *Member, IEEE*, and Jiachen Yang, *Senior Member, IEEE*

***Abstract*—The unmanned aerial vehicles (UAVs) are competent to perform a variety of applications, possessing great potential and promise. The deep neural networks (DNN) technology has enabled the UAV-assisted paradigm, accelerated the construction of smart cities, and propelled the development of the Internet of Things (IoT). UAVs play an increasingly important role in various applications, such as surveillance, environmental monitoring, emergency rescue, and supplies delivery, for which a robust path-planning technique is the foundation and prerequisite. However, existing methods lack comprehensive consideration of the complicated urban environment and do not provide an overall assessment of the robustness and generalization. Meanwhile, due to the resource constraints and hardware limitations of UAVs, the complexity of deploying the network needs to be reduced. This article proposes a lightweight, reinforcement-learning-based (RL-based) real-time path-planning method for UAVs, adaptive SAC (ASAC) algorithm, which optimizing the training process, network architecture, and algorithmic models. First of all, we establish a framework of global training and local adaptation, where the structured environment model is constructed for interaction, and local dynamically varying information aids in improving generalization. Second, ASAC introduces a cross-layer connection approach that passes the original state information into the higher layers to avoid feature loss and improve learning efficiency. Finally, we propose an adaptive temperature coefficient, which flexibly adjusts the exploration probability of UAVs with the training phase and experience data accumulation. In addition, a series of comparison experiments have been conducted in conjunction with practical application requirements, and the results have fully proved the favorable superiority of ASAC.**

***Index Terms*—Internet of Things (IoT), path planning, real-time applications, reinforcement learning, tiny machine learning (ML), unmanned aerial vehicles (UAVs).**

I. Introduction

THE UNMANNED aerial vehicles (UAVs) are a kind of intelligent unmanned equipment, which are characterized by advantages, such as light weight, mobility, and flexibility. The concept of a UAV-assisted paradigm has emerged over the recent years, where UAVs can supplement the tasks of numerous applications [1], such as UAV and unmanned ground vehicle (UGV) coordination [2], data gathering [3], disaster management [4], and so on. At the same time, benefiting from the advanced machine learning (ML) technology, people are deploying various neural network models and building UAV Internet of Things (IoT) networks to accomplish more complex application tasks [5], [6].

In particular, the construction of smart cities attracts a lot of attention, and UAVs play an essential role. For example, UAVs can be applied to urban surveillance, environmental monitoring, emergency rescue, and material delivery [7], [8]. In urban scenarios, the UAV-assisted paradigm helps to enhance the safe construction of infrastructure, promote the efficient operation of life services, and facilitate the living experience of citizens. Path planning, as a basic function of the intelligent unmanned equipment, is a prerequisite for numerous missions. Therefore, this article focuses on the urgent problem of UAV path planning for urban applications.

Research on autonomous, real time, and lightweight path planning method is of great significance to the development of UAVs [9]. Generally, the problem can be divided into two, global and local path planning. The former is static planning that acquires all the environmental information in advance; the latter is dynamic planning that utilizes real-time environmental information fed back by sensors. The existing methods for UAV path planning can be categorized into three categories: 1) the traditional methods; 2) the heuristic methods; and 3) the deep-learning-based methods. The traditional methods are based on random sampling or global search. Representative methods include A*, Dijkstra, rapid-exploration random tree (RRT), and RRT*. However, the shortcomings of such algorithms lie in the longer search time, higher computational consumption, and poorer real-time performance. Thus, they are commonly used for global path planning and difficult to adapt to the dynamic environment. The heuristic methods are based on intuitive and empirical approaches to finding feasible solutions, and common algorithms draw on the evolutionary mechanisms or organization of biological populations, e.g., genetic algorithm (GA), ant colony optimization (ACO), particle swarm optimization (PSO), etc. Their performance is greatly dependent on human experience and manual functions,

Manuscript received 4 November 2023; revised 4 December 2023 and 25 December 2023; accepted 3 January 2024. Date of publication 5 January 2024; date of current version 7 June 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62271345; in part by the China Postdoctoral Science Foundation under Grant 2021M702441; and in part by the Joint Fund of Ministry of Education for Equipment Preresearch under Grant 8091B032254. *(Corresponding author: Jingyi He.)*

The authors are with the School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China (e-mail: ximeng@tju.edu.cn; daihuiao@tju.edu.cn; seie_hjy@tju.edu.cn; li_wenjie@tju.edu.cn; Wen_Jiabao@tju.edu.cn; xs611@tju.edu.cn; yangjiachen@tju.edu.cn).

Digital Object Identifier 10.1109/JIOT.2024.3350525

which lacks the flexibility to ensure an optimal solution. The deep-learning-based methods utilize deep neural networks (DNN) for feature extraction to achieve end-to-end learning. For example, artificial neural network-based, convolutional neural network-based, and reinforcement-learning-based (RL-based) approaches. Compared to the previous two, such deep-learning-based methods demonstrate superiority: high flexibility, strong adaptability, and applicable to both global and local path-planning problems.

Constrained by the hardware conditions of UAVs, their computational resources and capabilities are limited. Complex neural networks and real-time application tasks are still considerable challenges. Utilizing the artificial intelligence technology to improve the autonomous decision making and control capabilities of UAVs has become the key. Although, the deep-learning-based methods have incomparable advantages. The development and application face bottlenecks in the resource-constrained UAVs. First, high cost of training. The end-to-end network relies on the number of samples, training rounds, etc., which it is time-consuming. Second, heavy computational overhead. DNN models have high complexity, leading to a contradiction between network size and learning effectiveness, accuracy and latency. Third, strong interference and uncertainty. There are many disturbances in the operating environment. For example, winds at high altitude make deviation from the preset trajectory, unexpected events lead to collision, and so on. Consequently, targeted research is urgently needed to reduce training overhead, alleviate computational constraints of UAVs, and improve algorithmic robustness.

To overcome these challenges, we propose a lightweight RL-based UAV path-planning method, adaptive SAC (ASAC) , which integrally enhances the performance by optimizing the training process, modifying the network structure and refining the algorithmic model. First of all, we establish a framework of global training and local adaptation. A structured environment model is established based on globally known information for interaction and training of planning models, followed by local dynamic varying state to assist in adaptive adjustment. Subsequently, we reduce the burden on computational resources by modifying the network structure, which utilizes just 3–5 layers of multilayer perceptron (MLP) and the cross-layer connection to feed the underlying information into the upper layers in order to fully extract effective feature. Finally, we propose an adaptive temperature coefficient, in which the UAV attenuates the probability of exploration of the environment as interaction rounds and experience accumulate. The main contributions of this article are as follows.

1) A global training and local adaptive framework is proposed, which rationally utilizes effective information, including globally known maps and unknown dynamic contingencies, and this structured training process can reduce the time cost and improve the efficiency.
2) We design a lightweight cross-layer connection for the model network, which ensures the feature extraction performance and learning effect of the DNN, importantly alleviates the limitation of computational resources of tiny devices.

Fig. 1. Illustration of a UAV working in the urban areas.

3) We propose an adaptive temperature coefficient tuning approach, which designs the decay function to adjust the exploration probability of the UAV according to the external conditions, which greatly improves the performance of the algorithm.

The remainder of this article is organized as follows. In Section II, we introduce some related works of UAV path planning. Then, the method is elaborated in Section III. In Section IV, a series of experiments were conducted and analyzed. Finally, we summarize our work and discuss the future directions in Section V.

II. Related Work

Path planning is the basis and prerequisite for UAVs to perform real-time missions, and the requirements vary from application to application. First, different scenarios have diverse demands on technology. Unstructured environments, such as woods or forests require high robustness and immunity to interference [10]. Structured scenarios, such as cities and villages with dynamic obstacles need to focus on the collision avoidance ability [11], [12]. Second, different tasks require varying performance of the algorithms. Surveillance, search and rescue, etc., require planning paths that cover as large an areas as possible [13], [14]. UAV-UGV cooperation requires consideration of elements, such as collaboration efficiency and communication range [15], [16]. Emergency situations have shorter planning times and path lengths [17]. UAVs with limited resources need to consider energy consumption [18]. In summary, different scenarios require the development of targeted approaches, and this article focuses on the impact of dynamic obstacles, wind noise, and other environmental disturbances on path planning in complex urban areas as Fig. 1.

The research on UAV path-planning algorithms has received extensive attention from both academia and industry [19], [20]. As mentioned earlier, traditional or heuristic algorithms are difficult to meet the requirements of practical applications. As a result, many researchers have improved the basic algorithms. The main problems of traditional algorithms

are long search times and poor path performance. Li et al. [21] improved the A* algorithm in terms of urban airspace constraints and UAV performance limitations to improve delivery efficiency and cost. Hohmann et al. [22] integrated the Dijkstra search and meta-heuristics to improve the accuracy of the algorithm. Chang et al. [23] introduced a spatial skeleton extraction method into RRT to accelerate the search process and smooth the path. The Voronoi-based path generation method utilized Voronoi diagrams to further reduces the time complexity [24].

Improvements for heuristic algorithms aim at overcoming the problem of local optimum. For example, Yu et al. [25] combined the simulated annealing algorithm with the PSO to improve the updating strategy of the globally optimal solution, effectively avoiding falling into local convergence. Similarly, Wan et al. [26] improved the ACO by optimizing the search direction and neighborhood mechanism. Wu et al. [15] hybridized the estimation of the distribution algorithm and GA to enhance the search efficiency.

Among the deep-learning-based path-planning methods, RL-based algorithms are one of the most remarkable approaches that have been developed to address various aspects of UAV path planning. Depending on the underlying algorithm, RL-based algorithms can be further categorized into value-based and actor–critic-based approaches. The first category creates a value network, known as a Q-network, to aid in the development of strategy. For example, Messaoudi et al. [2] employed a multiagent deep Q-network (MADQN) for the energy-constrained UAV in data collection, which can significantly reduce the Age of Information (AoI) of IoT devices. Wei et al. [27] built a deep reinforcement learning framework and improved the deep Q-learning network (DQN) algorithm to solve the Nondeterministic Polynomial-Hard problem in planning. Wang et al. [28] utilized the Dueling double deep Q-network (D3QN) algorithm to plan the UAV motion strategy and was able to coping with the complex scenarios with dynamic obstacles in the absence of a priori environmental knowledge. The second category adds a policy network, which outputs decisions directly, and this category is usually applicable to continuous tasks. For example, Samir et al. [29] utilized the deep deterministic policy gradient (DDPG) algorithm to effectively address the deployment and scheduling strategies of UAVs in intelligent transportation. Wang et al. [30] improved the Twin-Delayed DDPG algorithm to continuously and adaptively learn to adjust the motion strategy of UAVs.

In summary, with the development of the reinforcement learning algorithm, its scope of application has been extended from simple to complex scenarios, capable of solving sophisticated decision-making tasks, including UAV-UGV collaboration, data collection, emergency rescue, and more. Existing methods are summarized in Table I. The analysis shows that the existing methods do not comprehensively consider the complexity of the urban environment, including buildings, obstacles, and other physical disturbances. Meanwhile, research should be conducted to improve the robustness and generalization of the algorithm to adapt to the complex environment. Therefore, the aim of this article is to carry out a comprehensive and fundamental study of RL-based path-planning algorithms for UAV-assisted paradigm in the complex urban scenarios in order to serve a wider range of assignments.

III. Methodology

A. UAV Model

UAV is usually regarded as a uniform rigid body with 3-D and six-degree-of-freedom motion. Its motion simulation is based on the Newton–Euler Equations to complete the conversion of force F and moment M to position $\mathbf{p}$ and attitude Θ. In this article, a quadcopter UAV is analyzed as an example in Fig. 2.

The dynamics model converts the combined external force and moment on the UAV into velocity and angular velocity. The coordinate system of the Earth is $O_e - X_eY_eZ_e$ and the coordinate system of the UAV body is $O_b - X_bY_bZ_b$. Let the current position of the UAV be $\mathbf{p} = [x, y, z]^T$, velocity be $\mathbf{v} = [\dot{x}, \dot{y}, \dot{z}]^T$, acceleration be $\dot{\mathbf{v}} = [\ddot{x}, \ddot{y}, \ddot{z}]^T$, attitude be $\Theta = [\phi, \theta, \psi]^T$, angular velocity $\omega = [p, q, r]^T$, and angular acceleration $\dot{\omega} = [\dot{p}, \dot{q}, \dot{r}]^T$. The following relationship can be obtained from Newton's second law:

$$\dot{\mathbf{v}} = g - R_b^e \frac{F}{m}. \tag{1}$$

g is the gravitational acceleration. R_b^e is the rotation transformation matrix of the coordinate system, which transforms the motion of the UAV from the body coordinate system to the Earth. $F = \sum_{i=1}^{4} F_i$ is the overall tension force. The following equation can be obtained from Euler's equation:

$$\mathbf{J}\dot{\omega} = -\omega \times \mathbf{J}\omega + \mathbf{G}_a + \tau. \tag{2}$$

$\mathbf{G}_a$ is the gyroscopic moment, which is generally negligible. $\mathbf{J} = \text{diag}(J_x, J_y, J_z)$ is the inertia matrix. τ is the joint external moment

The kinematics model calculate the position and attitude of the UAV based on velocity and angular velocity

$$\dot{\mathbf{p}} = \mathbf{v} \tag{3}$$

$$\dot{\Theta} = \mathbf{W}\omega. \tag{4}$$

$\mathbf{W}$ is the conversion matrix of attitude velocity to body angular velocity. The above is the motion model of the UAV, based on which the subsequent analysis is carried out.

B. Reinforcement Learning Model

Reinforcement learning simulates the human learning process. Specifically, the problem is abstracted into two parts, the Environment and the Agent model. Design the interaction between the two components and utilize the interaction data to learn anthropomorphic skills such as decision making. The details of the interaction scheme are designed as follows.

1) Environment Model: This article targets the study of tiny UAV devices working in structured urban scenarios, thus constructing a complex urban spatial environment with a scale of $50 \times 50 \times 50$. In order to construct complex structured buildings, we divide the plane in units of 10 so that each side is divided into 5. The plane contains 5×5 subspaces, each of which randomly generates buildings with different

TABLE I
SUMMARY OF UAV PATH-PLANNING METHODS

Ref	Objective	Categories	Method	Advantages	Drawback
Li *et al.* [21]	Plan path, and design evaluation programs for urban logistics UAVs	traditional algorithm	Improved A* algorithm	Consider urban airspace constraints and performance limitations	No consideration of real-time and obstacle avoidance
Hohmann *et al.* [22]	Multi-objective path planning in 3D space	traditional algorithm	Improved Dijkstra algorithm	Consider system failure, signal disturbance, and energy consumption	Suitable for grid-based objectives, and operation space is limited
Chang *et al.* [23]	Plan path from start to the goal and smooth the trajectory	traditional algorithm	Improved RRT algorithm	Reduce path search time and improve efficiency	No consideration of dynamic obstacles
Jensen-Nau *et al.* [24]	For energy-limited UAVs, and increase coverage area	traditional algorithm	Improved Voronoi-based algorithm	Better balance of runtime and optimality	Accurate energy modeling is needed
Yu *et al.* [25]	Collision-free path planning in complex mountainous area	heuristic algorithm	Improved PSO algorithm	Faster convergence, higher robustness	Other environmental disturbances are not addressed (e.g. wind)
Wan *et al.* [26]	Fast, collision-free routes for disaster emergency response	heuristic algorithm	Improved ACO algorithm	Improved global and local search capabilities	Lack of constraints on actual flight conditions
Messaoudi *et al.* [2]	Data collection for energy-constrained UAVs in unknown IoT environment	RL-based algorithm	Improved DQN algorithm	Minimize the average AoI of IoT devices, and optimize trajectory	Continuous state-action spaces have not yet been discussed
Wei *et al.* [27]	UAV serves a complex area with multiple obstacles and dependent tasks	RL-based algorithm	Improved DQN algorithm	Higher success rates, more task completions, and less latency	Obstacles are regular and homogeneous
Wang *et al.* [28]	Data collection in non-cooperative scenarios	RL-based algorithm	Improved D3QN algorithm	Adaptation to the scenarios without priori information	No consideration of jamming attacks
Samir *et al.* [29]	Jointly optimize the trajectory and find scheduling policy	RL-based algorithm	Improved DDPG algorithm	Minimize the AoI under minimum the throughput constraints	No consideration of more precise physical parameters
Wang *et al.* [30]	Path planning in practical 3D urban with imperfect channel state information	RL-based algorithm	Improved TD3 algorithm	Merged pheromone enhances adaptive capacity and minimizes time	No consideration of obstacles and physical constraints
Our Method	Path planning in 3D urban scenarios with multiple obstacles and physical distractions	RL-based algorithm	Improved SAC algorithm	Higher success rate, robustness and generalization in diverse and complex scenarios	Accurate energy modeling is needed

shapes and heights. The UAV travels between these buildings to accomplish specific tasks, such as distribution, monitoring, and filming. In addition, randomly dynamic obstacles and wind noise are added to investigate the robustness, flexibility, and generalization of algorithms.

The most important element in the Environment model is the reward function, which is the feedback that the agent receives for taking different action strategies. It is intuitively understood that good strategies should receive high rewards, and conversely, bad actions should receive low rewards (i.e., punishment). The reward function is directly related to the performance of the agent, which is designed as follows:

$$R = r_1 + r_2 + r_3 + r_4. \tag{5}$$

r_1 is the distance reward component, it evaluate the distance between the position and the goal $\mathbf{p}_g$, aiming to guide the UAV closer to the goal. At time t_i, the UAV is located at $\mathbf{p_i}$, and r_1 is calculated as follows:

$$\Delta \mathrm{p}_i = \left\| \mathbf{p}_i - \mathbf{p}_g \right\| \tag{6}$$

$$r_1 = \Delta \mathrm{p}_{i+1} - \Delta \mathrm{p}_i. \tag{7}$$

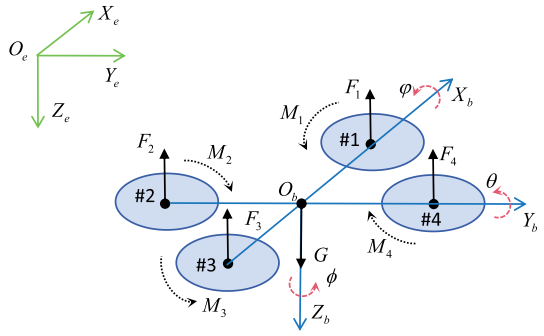

Fig. 2. Model of a quadrotor UAV.

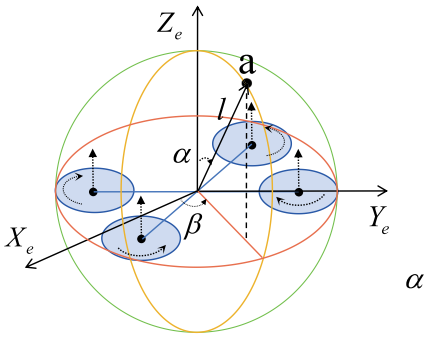

Fig. 3. Illustration of the action space.

r_2 is the collision penalty to avoid failures caused by external interference, once a collision with a building or obstacle occurs, the penalty will be given as follows:

$$r_2 = \begin{cases} -5, & \text{if collision} \\ 0, & \text{else} \end{cases} \tag{8}$$

$r_3 = -1$ is the time feedback, which is a constant value, meaning that every movement before reaching the goal point will be penalized. r_4 is the final goal feedback, when the goal point is reached, a bonus is

$$r_4 = \begin{cases} 10^2, & \text{if reach goal} \\ 0, & \text{else.} \end{cases} \tag{9}$$

In summary, guided by the above rewards, the agent will be able to plan the optimal collision-free path.

2) Agent Model: The most critical elements of the Agent model are the state space and the action space. State space refers to the information that the UAV can observe and make decisions based on. The state space of the UAV is $S = [O_1, O_2, O_3]^T$, including three components. The first O_1 represents its own location information $\mathbf{p} = [x, y, z]^T$. O_2 is the environment information, which contains the information about wind ($[w_x, w_y, w_z]^T$, values of in the direction of the three coordinate axes), buildings in the adjacent four subspaces ($[x_i, y_i, a_i, b_i, h_i]^T$, $i = 1, 2, 3, 4$, where (x_i, y_i) is the location and (a_i, b_i, h_i) is the shape of length, width, and height), and obstacles ($[x_o, y_o, z_o]^T$, the coordinates). O_3 is the goal position $\mathbf{p}_g = [x_g, y_g, z_g]^T$. A clearer description of the state space can be visualized in the purple box in Fig. 4.

In order to facilitate the study of the algorithm, following the previous setup [31], [32], the UAV is abstracted and reduced to a particle. Its action space is $A = [\alpha, \beta, l]^T$, where $\alpha \in [0, \pi]$ and $\beta \in [0, 2\pi]$ are attitude angles, l is a constant, as Fig. 3.

C. ASAC Algorithm

Our algorithm is divided into two phases: 1) interaction and 2) update; and contains four components: 1) Environment; 2) Actor; 3) Critics; and 4) Experience Buffer, as Fig. 4.

1) Interaction Phases: The Environment model gives state s, reward r, and done indicator d based on the Agent's action a

$$s, r, d \leftarrow T(a) \tag{10}$$

where $T(a)$ is the state transfer function of the Environment model. Actor in the Agent model receives the state information from the Environment model and output action

$$a, \log(P(a)) \leftarrow F(s; \vartheta) \tag{11}$$

where $F(s; \vartheta)$ is the neural network of Actor; θ is the network parameter; and $\log(P(a))$ is the logarithm of the probability of action a. Experience Buffer $\mathbb{E}$ stores the historical data in batches, called experience

$$\left(s, s', a, r, d\right) \rightarrow \mathbb{E} \tag{12}$$

where s' is the next state and (s, s', a, r, d) is a set of experience. The above steps are repeated during the Interaction Phases, as shown in the blue box in Fig. 4, forming a closed loop guided by the thick gray arrow.

2) Update Phases: There is an Actor and four Critics in the Agent model, and historical data from the Experience Buffer is extracted to update their neural networks. The specific details are shown in the black box and indicated by colored thin arrows in Fig. 4.

The Actor network $F(S; \vartheta)$ learns strategies and outputs the UAV's actions. Its updated objective is to master the preferred strategy that help the UAV form a collision-free route to the goal point. First of all, a batch of experiences is drawn and the Actor network gives the action prediction $\widehat{a}$ and the corresponding probability

$$\widehat{a}, \log\Big(P(\widehat{a})\Big) \leftarrow F(s; \vartheta). \tag{13}$$

Then, two Critic networks are $Q_1(\mathrm{S}, A; \varpi_1)$, $Q_2(S, A; \varpi_2)$, and ϖ_i is the network parameter. They evaluate the action to get the state-action value $\widehat{q}_i$

$$\begin{aligned} \widehat{q}_1 &\leftarrow Q_1\Big(s, \widehat{a}; \varpi_1\Big) \\ \widehat{q}_2 &\leftarrow Q_2\Big(s, \widehat{a}; \varpi_2\Big). \end{aligned} \tag{14}$$

Finally, the loss function of the Actor network is

$$\nabla_\vartheta = \lambda \log(P(a)) - \min\Big(\widehat{q}_1, \widehat{q}_2\Big) \tag{15}$$

where λ is the temperature coefficient. Its value changes adaptively as the UAV is explored, as (23).

There are four critic networks, Critic#1, Critic#2, Critic#1-Target, and Critic#2-Target, serving different purposes. The first two are responsible for evaluating the value of the actor's

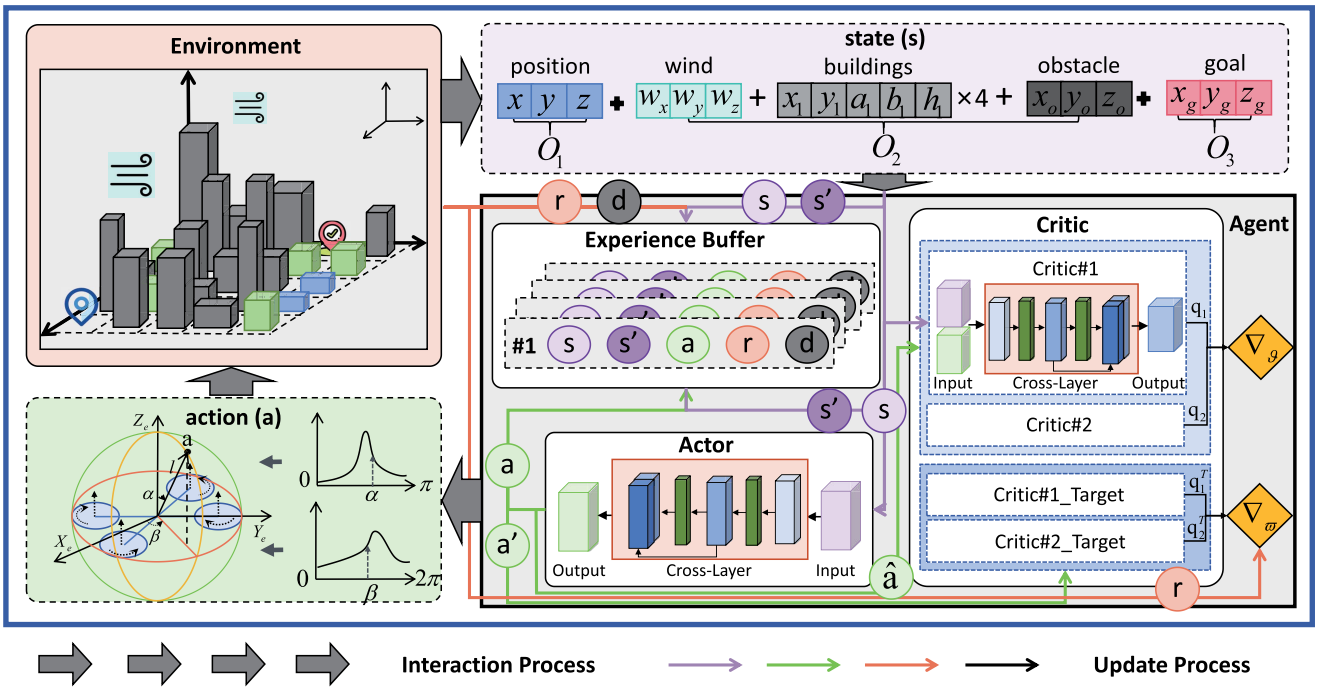

Fig. 4. Pipeline of our method, which consists of two processes, the interaction phase and the update phase. The interaction phase refers to the continuous interaction between the agent and the environment to generate experience, which is stored in the experience buffer, and this process is represented using the blue boxes and gray thick arrows loops. The update phase refers to the process in which the agent extracts experience to update it to each of the five neural networks, Actor, Critic#1, Critic#2, Critic#1-Target, and Critic#2-Target, and this process is represented by the black boxes and colored thin arrows.

action, so they are updated with the aim of improving the accuracy of the value estimation and avoiding an overestimation. The latter two fix the value of the neural network and serve as the target, avoiding the jitter caused by frequent updates. Specifically as follows, at first, the Actor gives the next action a' about to be taken:

$$a', \log\big(P(a')\big) \leftarrow F\big(s'; \vartheta\big). \tag{16}$$

The two target networks $Q_1^T(S, A; \varpi_1^T)$ and $Q_2^T(S, A; \varpi_2^T)$ give the estimations of the next state-action (s', a')

$$\begin{aligned} q_1^T &\leftarrow Q_1^T\big(s', a'; \varpi_1^T\big) \\ q_2^T &\leftarrow Q_2^T\big(s', a'; \varpi_2^T\big). \end{aligned} \tag{17}$$

Subsequently, together with the reward feedback from the Environment, the value for the next state-action can be obtained

$$q = r + \gamma\big(\min(\mathrm{q}_1^T, q_2^T) + \lambda \log(P(a'))\big) \tag{18}$$

where $\gamma = 0.99$ is a discount factor. Critic#1 and Critic#2 output their estimations of the current state-action

$$\begin{aligned} q_1 &\leftarrow Q_1(s, a; \varpi_1) \\ q_2 &\leftarrow Q_2(s, a; \varpi_2). \end{aligned} \tag{19}$$

Finally, the loss function for Critic#1 and Critic#2 is

$$\nabla_{\varpi_i} = \frac{1}{B} \sum_{j=1}^{B} \big((q_i - q)_j\big)^2, i = 1, 2. \tag{20}$$

$B = 2^8$ is a batch of experience. The two target networks periodically copy the corresponding parameters for soft updates

$$Q_i^T\big(S, A; \varpi_i^T\big) = \tau Q_i(S, A; \varpi_i) + (1 - \tau) Q_i^T\big(S, A; \varpi_i^T\big) \tag{21}$$

where $\tau = 5 \times 10^{-3}$ is the parameter for soft update.

Limited by the computational resources of the miniaturized device, the five neural networks of Actor and Critics could not be too large. Therefore, in order to ensure sufficiently adequate learning effect, our method optimizes the traditional reinforcement learning networks and designs a cross-layer connection structure. Take the Actor network $F(S; \vartheta)$ as an example. ϑ is the parameter matrix, and the kth layer of the neural network has n^k neurons, the $k+1$th layer with n^{k+1} neurons. ϑ^{k+1} is obtained from ϑ^{k+1} by calculating the weight matrix δ^k and bias matrix ς^k. The relationship is as follows:

$$\begin{bmatrix} \vartheta_1^{k+1} \\ \vartheta_2^{k+1} \\ \vdots \\ \vartheta_{n^{k+1}}^{k+1} \end{bmatrix} = \begin{bmatrix} \delta_{11}^k & \delta_{12}^k & \cdots & \delta_{1n^k}^k \\ \delta_{21}^k & \delta_{22}^k & \cdots & \delta_{2n^k}^k \\ \vdots & \vdots & \vdots & \vdots \\ \delta_{n^{k+1}1}^k & \delta_{n^{k+1}1}^k & \cdots & \delta_{n^{k+1}n^k}^k \end{bmatrix} \begin{bmatrix} \vartheta_1^k \\ \vartheta_2^k \\ \vdots \\ \vartheta_{n^k} \end{bmatrix} + \begin{bmatrix} \varsigma_1^k \\ \varsigma_2^k \\ \vdots \\ \varsigma_{n^k}^k \end{bmatrix} + \begin{bmatrix} \vartheta_1^k \\ \vartheta_2^k \\ \vdots \\ \vartheta_{n^k} \end{bmatrix} = \big(\delta^k + 1\big)\vartheta^k + \varsigma^k. \tag{22}$$

The advantages of the cross-layer connection structure are as follows.

1) The loss of important features is avoided, keeping the original information intact and enriching the input.

2) Bringing new information to the last layer of the network prevents overfitting and enhances robustness to some extent.
3) Element-level operations minimize the amount of data and computation while ensuring increased information and improved learning performance.

In (15) and (18), the temperature coefficient λ serves to increase the exploration during the interaction, with larger values encouraging more actions to be attempted. This has the potential to find better strategies and plan preferable paths. ASAC has improved the traditional setting of λ by designing a decay function which is more flexible and adaptable

$$\lambda = \lambda_0 + (1 - \lambda_0)e^{\frac{-n}{\iota}} \tag{23}$$

where $\lambda_0 = 10^{-1}$ is the minimum value; n is the global step counter, increasing with training episodes; and $\iota = 10^4$ is the attenuation factor. During the initial stage, the environment is unknown, and the larger λ encourages the UAV to explore and master the skills through continuous trial and error. On the contrary, in the later stage, the larger exploration is a waste of computing resources. Equation (23), which is regulated with the training phase in the presence of n, is able to tackle this problem reasonably.

The details of the algorithm and parameter definitions are shown in Algorithm 1 and Table II.

Algorithm 1: ASAC Algorithm

initial global step and episode counter $n = m = 0$
for $m = 0$ *to* M **do**
  $n = n + 1$
  if $n <= N$ **then**
    randomly sampling: $a \leftarrow A$
  else
    Actor outputs action: $a, \log(P(a)) \leftarrow F(s; \vartheta)$
  end
  Environment back: $s', r, d \leftarrow T(a)$
  if d **then**
    $m = m + 1$
  end
  Experience Buffer store: $(s, s', a, r, d) \rightarrow \mathbb{E}$
  if $n > N$ **then**
    extract batches of experience for updating: $B(s, s', a, r, d) \leftarrow \mathbb{E}$
    Actor outputs the next action: $a', \log(P(a')) \leftarrow F(s'; \vartheta)$
    two target networks output the estimations $\mathrm{q}_1^T \leftarrow Q_1^T(\mathrm{s}', a'; \varpi_1^T)$, $\mathrm{q}_2^T \leftarrow Q_2^T(\mathrm{s}', a'; \varpi_2^T)$
    the value of the next state-action: $\mathrm{q} = r + \gamma(\min(\mathrm{q}_1^T, \mathrm{q}_2^T) + \lambda \log(P(a')))$
    losses for Critic#1 and Critic#2 networks: $\nabla_{\varpi_i} = \frac{1}{B}\sum_{j=1}^{B}((Q_i(s, a; \varpi_i) - \mathrm{q})_j)^2, i = 1, 2$
    while n *mod* $f_a == 0$ **do**
      Actor outputs the action: $\widehat{a}, \log(P(\widehat{a})) \leftarrow F(s; \vartheta)$
      two Critic networks output the estimations: $\widehat{\mathrm{q}}_1 \leftarrow Q_1(s, \widehat{a}; \varpi_1)$, $\widehat{\mathrm{q}}_2 \leftarrow Q_2(s, \widehat{a}; \varpi_2)$
      loss for Actor network: $\nabla_\vartheta = \lambda \log(P(a)) - \min(\widehat{\mathrm{q}}_1, \widehat{\mathrm{q}}_2)$
    end
    while n *mod* $f_c == 0$ **do**
      soft update the two target networks: $Q_i^T(\mathrm{S}, A; \varpi_i^T) = \tau Q_i(\mathrm{S}, A; \varpi_i) + (1 - \tau)Q_i^T(\mathrm{S}, A; \varpi_i^T), i = 1, 2$
    end
  end
end

IV. Experiments

A. Basic Settings

The experiments are carried out in the $50 \times 50 \times 50$ structured 3-D simulated urban environment, in which the location, number, and shape of the buildings can be randomly varied. In addition, random dynamic obstacles are placed between the buildings and wind noise interference is added at high altitude. A wide variety of urban scenarios are constituted. We compare several mainstream reinforcement learning algorithms, TD3, Proximal policy optimization (PPO), and soft actor–critic (SAC), which keep the same setup with 10^3 total episodes, a 5-layer MLP, 2^8 neurons per layer, and 3×10^{-4} for Adam policy optimizer. The comparative indicators are described in the following.

1) *Reward:* It is the UAV that gets feedback from the Environment model and reflects the performance of the algorithm during the training process. Higher value indicates better performance.
2) *Length:* It is the length of the path generated in each episode. For ease of representation, the units of the grid are used uniformly. When the UAV successfully reaches the goal, a smaller value indicates a shorter path.
3) *Step:* It refers to the number of times the UAV interacts with the Environment model in each episode. The maximum value for an episode is 10^3. The smaller the value, the shorter the planning time consumption.
4) *Number of Collisions:* It tracks the number of times that the UAV collides with buildings or obstacles during training. When the value is 0, it indicates a collision-free route.
5) *Success Rate:* The rate at which the trained model successfully reaches the goal without collision. This indicator is obtained by statistically counting 10^2 episodes.

B. Results and Analysis

In order to fulfill the practical requirements and to compare the performance of the algorithms in various aspects, we carry out five sets of comparative experiments in the context of complicated urban applications.

1) Validity Verification: The urban scenario is deterministic, and the UAV allows to plan the optimal path from

TABLE II
NOTATIONS

Name	Description
S	the state space
A	the action space
s, s'	the specific/next state
$a, a', \widehat{a}$	the specific/next/predicted action
$P(a)$	the probability of action a
r	reward
γ	discount factor
d	done indicator
$T(A)$	the Environment state transfer
$F(S;\vartheta)$	the Actor network
$Q_i(\mathrm{S}, A; \varpi_i)$	the Critic#i networks
$Q_i^T(\mathrm{S}, A; \varpi_i^T)$	the Critic#i-Target networks
M	total episodes
N	total randomized exploration steps
f_a	update frequency of Actor
f_t	update frequency of Critic#i-Target
B	batches of experience for updating
λ	temperature coefficient
τ	the parameter for soft update
ι	the attenuation factor
ϑ	the parameter for Actor network
ϖ_i, ϖ_i^T	the parameter for Critic networks

the starting $(1, 1, 6)$ to the goal $(49, 49, 6)$ with the globally known map. At this point, the situation is simpler and there are no additional dynamic obstacles. The experimental results are shown in Figs. 5 and 6. Fig. 6 is the result of the visualization of the planned paths, as mentioned before, the simulation space is a 3-D space of $50 \times 50 \times 50$ with randomly generated buildings of different sizes in the subspace.

Among these four algorithms, ASAC evidently demonstrates good superiority, TD3 has the weakest performance to solve this task. The results can be analyzed in the following aspects.

1) As a whole, ASAC converges faster and is able to learn to plan collision-free paths after about 10^2 episodes of training. It stabilizes at the highest reward value, indicating the best-planned path. In addition, ASAC also exhibits good stability and it has less curve jitter throughout the training session.
2) In Fig. 5(b), TD3 has higher values with constant jittering, and in Fig. 5(c), its Step value is always 10^3. These phenomena indicate that it keeps exploring in each episode, searching for a path to reach the goal, but never managed to find a collision-free route. The planned route is shown in Fig. 6(a).
3) In Fig. 5(b), PPO starts out with the shortest Length, but with the Step of 10^3 in Fig. 5(c), which occurs because the PPO hunkers down and circles around the starting point without actively exploring as Fig. 6(b).

2) Goal Generalization: In view of the fact that the goal may move its position due to unexpected situations. Therefore, we further investigate the goal generalization ability. In this set of comparisons, the starting point is fixed and the goal point moves randomly. The results are plotted in Fig. 7.

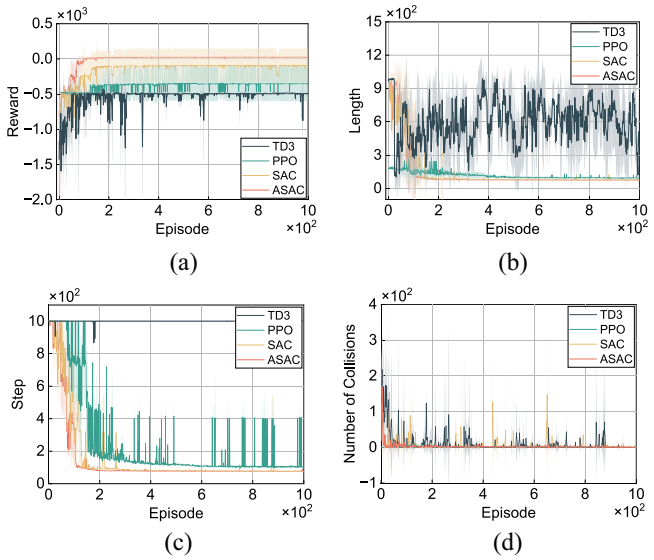

Fig. 5. Results of validity verification experiments. (a) Reward. (b) Length. (c) Step. (d) Number of collisions

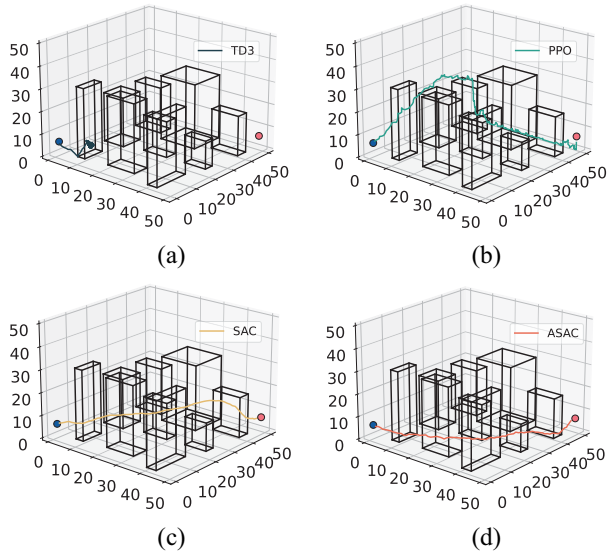

Fig. 6. Visualization of the path results. (a) TD3. (b) PPO. (c) SAC. (d) ASAC.

Compared to the first experiment, it was much more difficult to train due to the shifting of the goal point in each episode, such that TD3 and PPO both failed. The results of this set of experiments are characterized by the following.

1) In general, the training curves in Fig. 7(a)–(c) are more jittery than that in Fig. 5, due to the uncertainty of the goal point.
2) In this complex task, the gap between the performance of SAC and ASAC widens. The latter generalizes significantly better than the former, finding better paths more efficiently, with higher Reward, shorter Length, and fewer Step.
3) Since not every episode is able to reach the goal, the Success Rate becomes a more concerned indicator at this point, which of ASAC is obviously higher than that of the other algorithms.

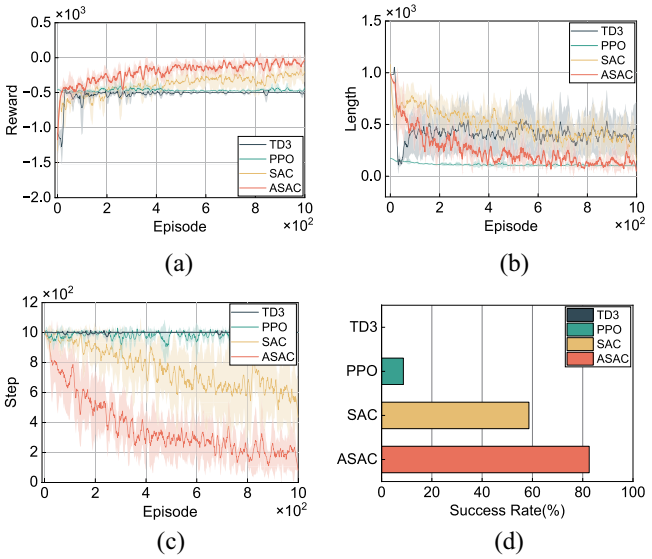

Fig. 7. Results of goal generalization experiments. (a) Reward. (b) Length. (c) Step. (d) Success Rate.

3) Scenario Generalization: In more challenging applications, the UAV does not know the global map, and it is only able to perform path planning based on limited goal point information. This generalization capability has a more substantial practical benefit, meaning that the UAV is able to work in arbitrary environments. Thus, in this set of comparison experiments, the urban scenario is changing in each different episode. Due to the escalation of the task, the requirement for algorithm generalization has become higher, at which point none of the other three algorithms can successfully reach the goal point, and only our method is able to cope with this challenge. Fig. 8 has a meaning similar to Fig. 6 and shows a visualization of ASAC's planning results in a number of different urban scenarios.

4) Interference of Obstacles: In crowded urban areas, there are numerous unpredictable situations, and one of the factors that cannot be ignored is dynamic obstacles. Randomly appearing obstacles pose a greater challenge to the robustness of the algorithm. To investigate the effect of this factor, dynamic obstacles are randomly added to each episode in this set of comparison experiments. The experimental results are shown in Fig. 9.

Since randomly appearing obstacles lead to common collisions in each episode, the comparison of Length and Step is not very significant, and we focus on the two indicators, Reward and Success Rate. It can be seen that:

1) only ASAC is able to solve this assignment, and it manages to avoid the dynamic obstacles and reach the goal point, thus ending up with a positive reward;
2) the phenomenon of TD3's lowest reward means that it hovers around the starting point. The reason is that its ability to explore is too low, and thus the cumulative penalties for each step of r_3 make cause the smallest value;
3) the Success Rate of ASAC is much higher than that of the other algorithms, even up to twice that of SAC and four times that of PPO.

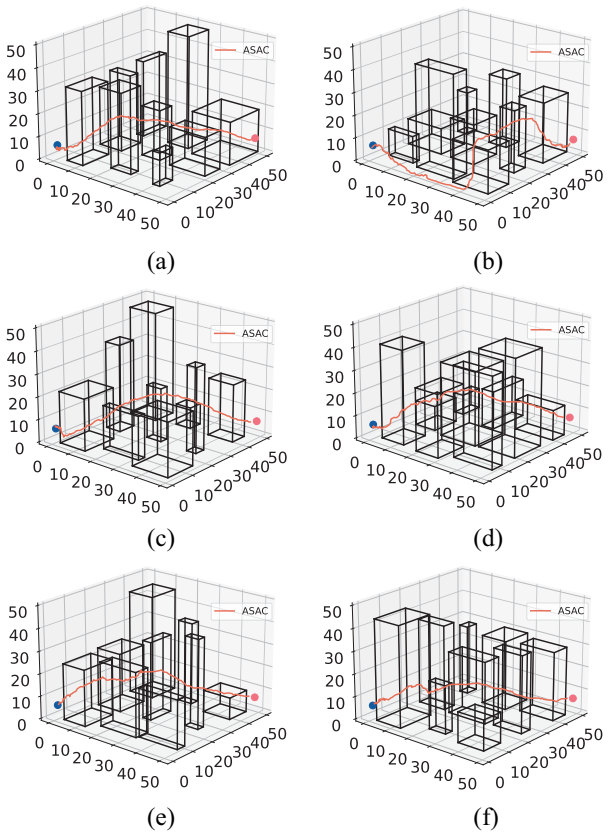

Fig. 8. Visualization results of some urban scenarios. (a) Scenario a. (b) Scenario b. (c) Scenario c. (d) Scenario d. (e) Scenario e. (f) Scenario f.

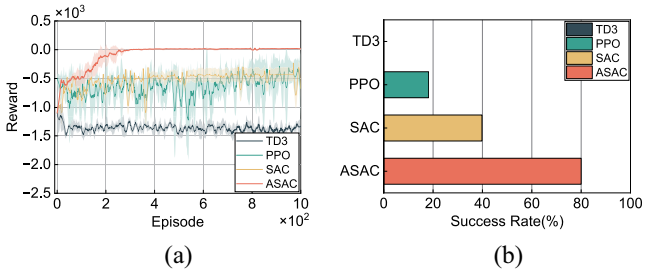

Fig. 9. Results of dynamic obstacles experiments. (a) Reward. (b) Success rate.

5) Interference of Wind Noise: The motion of the UAV is affected by wind and the disturbance is uncertain. A well-developed algorithm should be robust and stable enough to withstand the impact of wind. The stronger the wind, the more intense the interference. Therefore, this set of experiments compares the ability of the algorithms to cope with winds of different strengths. Since the experimental results are similar when the parameters are varied over a small range. Therefore, the range is expanded from 0.01 to 0.4 and a multiplicative growth setting is used to reflect the significant changes. The performance is graphed in Fig. 10. It can be seen that wind noise has a significant impact on the movement of the UAV in the following aspects.

1) Although TD3 is never able to reach the goal point, as indicated by the Step of 10^3 in Fig. 10(b), it had the longest Length at 0.01, meaning that it is able to hard to

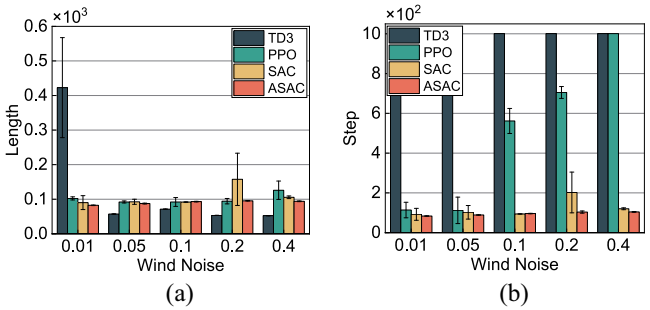

Fig. 10. Results of wind noise experiments. (a) Length. (b) Step.

explore. In other cases, however, the uncertainty of the environment prevented it from being explored, resulting in lower Length.

2) In Fig. 10(b) as the intensity increases, PPO consumes more and more Step up to the point where it fails completely at 0.4.
3) As a whole, ASAC has the best resistance to interference from wind noise, with the shortest Length, the fewest Steps, and the best stability.

In summary, from the above series of comparison experiments, it can be seen that: TD3 has the worst performance and is unable to complete the basic planning task. PPO has some generalization and ability to resist interference, but the success rate is lower. Although SAC is optimized for generalization and robustness, it cannot be applied to unknown and random urban scenarios. ASAC performs optimally in all aspects of robustness, generalization, and stability, and is able to cope with goal and scenario changes, adapt to the interference of dynamic obstacles and wind noise, and embody anthropomorphic intelligent decision making.

V. Conclusion and Discussion

In this article, we propose a lightweight RL-based real-time path-planning algorithm, ASAC, suitable for resource-constrained tiny device UAVs. First, we construct a framework of global training and local adaptation, where the environmental modeling and rational state representation can help accelerate skill acquisition and improve robustness to unexpected situations. Second, ASAC refines the network architecture by designing a cross-layer connection structures, which avoids the loss of valid features, enriches the information input, and also increases the generalization ability. Finally, ASAC proposes an adaptive temperature coefficient tuning approach, which flexibly reduces the exploration demand of UAV with the training phases and experience data accumulation. In addition, a series of comparative experiments are designed in conjunction with real-time application requirements, e.g., goal shifting, dynamic obstacles, wind noise interference, and so on. The results fully prove that ASAC has excellent and stable performance, and demonstrates advantages in flexibility, generalization, and robustness.

This article focuses on the requirements of path-planning techniques for UAV-assisted applications and proposes a real-time path-planning algorithm based on reinforcement learning. Our work still has some limitations, in order to facilitate the algorithmic research, it simplifies some constraints of actual flight conditions. In the future, we will work on more explorations in accurate energy modeling and continuous state-action space in the hope of further promoting the widespread applications of UAV assistance.

References

[1] B. Alzahrani, O. S. Oubbati, A. Barnawi, M. Atiquzzaman, and D. Alghazzawi, "Uav assistance paradigm: State-of-the-art in applications and challenges," *J. Netw. Comput. Appl.*, vol. 166, Sep. 2020, Art. no. 102706.

[2] K. Messaoudi, O. S. Oubbati, A. Rachedi, and T. Bendouma, "UAV-UGV-based system for AoI minimization in IoT networks," in *Proc. IEEE Int. Conf. Commun.*, 2023, pp. 4743–4748.

[3] K. Messaoudi, O. S. Oubbati, A. Rachedi, A. Lakas, T. Bendouma, and N. Chaib, "A survey of UAV-based data collection: Challenges, solutions and future perspectives," *J. Netw. Comput. Appl.*, vol. 216, Jul. 2023, Art. no. 103670.

[4] W. Wang, C. Fang, and T. Liu, "Multiperiod unmanned aerial vehicles path planning with dynamic emergency priorities for geohazards monitoring," *IEEE Trans. Ind. Informat.*, vol. 18, no. 12, pp. 8851–8859, Dec. 2022.

[5] N. Cheng et al., "AI for UAV-assisted IoT applications: A comprehensive review," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14438–14461, Aug. 2023.

[6] M. Adil, H. Song, S. Mastorakis, H. Abulkasim, A. Farouk, and Z. Jin, "UAV-assisted IoT applications, cybersecurity threats, AI-enabled solutions, open challenges with future research directions," *IEEE Trans. Intell. Veh.*, early access, Aug. 31, 2023, doi: 10.1109/TIV.2023.3309548.

[7] K. Zheng et al., "BRR-DQN: UAV path planning method for urban remote sensing images," in *Proc. China Autom. Congr. (CAC)*, 2021, pp. 6113–6117.

[8] Z. Pei, T. Fang, K. Weng, and W. Yi, "Urban on-demand delivery via autonomous aerial mobility: Formulation and exact algorithm," *IEEE Trans. Autom. Sci. Eng.*, vol. 20, no. 3, pp. 1675–1689, Jul. 2023.

[9] Z. Wei et al., "UAV-assisted data collection for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 15460–15483, Sep. 2022.

[10] L. Bartolomei, L. Teixeira, and M. Chli, "Fast multi-UAV decentralized exploration of forests," *IEEE Robot. Autom. Lett.*, vol. 8, no. 9, pp. 5576–5583, Sep. 2023.

[11] J. Li, Y. Xiong, and J. She, "UAV path planning for target coverage task in dynamic environment," *IEEE Internet Things J.*, vol. 10, no. 20, pp. 17734–17745, Oct. 2023.

[12] N. Bashir, S. Boudjit, and G. Dauphin, "A connectivity aware path planning for a fleet of UAVs in an urban environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 10, pp. 10537–10552, Oct. 2023.

[13] J. Chen, C. Du, Y. Zhang, P. Han, and W. Wei, "A clustering-based coverage path planning method for autonomous heterogeneous UAVs," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25546–25556, Dec. 2022.

[14] J. Xie and J. Chen, "Multiregional coverage path planning for multiple energy constrained UAVs," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17366–17381, Oct. 2022.

[15] Y. Wu, S. Wu, and X. Hu, "Cooperative path planning of UAVs & UGVs for a persistent surveillance task in urban environments," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4906–4919, Mar. 2021.

[16] J. Li et al., "A memetic path planning algorithm for unmanned air/ground vehicle cooperative detection systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 2724–2737, Oct. 2022.

[17] M. Li, S. He, and H. Li, "Minimizing mission completion time of UAVs by jointly optimizing the flight and data collection trajectory in UAV-enabled WSNs," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13498–13510, Aug. 2022.

[18] K. Liu and J. Zheng, "UAV trajectory optimization for time-constrained data collection in UAV-enabled environmental monitoring systems," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 24300–24314, Dec. 2022.

[19] M. Xi, J. Yang, J. Wen, H. Liu, Y. Li, and H. H. Song, "Comprehensive ocean information-enabled AUV path planning via reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17440–17451, Sep. 2022.

[20] J. Yang, J. Huo, M. Xi, J. He, Z. Li, and H. H. Song, "A time-saving path planning scheme for autonomous underwater vehicles with complex underwater conditions," *IEEE Internet Things J.*, vol. 10, no. 2, pp. 1001–1013, Jan. 2023.
[21] B. Li, H. Zhang, L. Zhang, D. Feng, and Y. Fei, "Research on path planning and evaluation method of urban logistics UAV," in *Proc. 3rd Int. Acad. Exchange Conf. Sci. Technol. Innov.(IAECST)*, 2021, pp. 1465–1470.
[22] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, "Three-dimensional urban path planning for aerial vehicles regarding many objectives," *IEEE Open J. Intell. Transp. Syst.*, vol. 4, pp. 639–652, 2023.
[23] J. Chang, N. Dong, D. Li, W. H. Ip, and K. L. Yung, "Skeleton extraction and greedy-algorithm-based path planning and its application in UAV trajectory tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, no. 6, pp. 4953–4964, Dec. 2022.
[24] K. R. Jensen-Nau, T. Hermans, and K. K. Leang, "Near-optimal area-coverage path planning of energy-constrained aerial robots with application in autonomous environmental monitoring," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 3, pp. 1453–1468, Jul. 2021.
[25] Z. Yu, Z. Si, X. Li, D. Wang, and H. Song, "A novel hybrid particle swarm optimization algorithm for path planning of UAVs," *IEEE Internet Things J.*, vol. 9, no. 22, pp. 22547–22558, Nov. 2022.
[26] Y. Wan, Y. Zhong, A. Ma, and L. Zhang, "An accurate UAV 3-D path planning method for disaster emergency response based on an improved multiobjective swarm intelligence algorithm," *IEEE Trans. Cybern.*, vol. 53, no. 4, pp. 2658–2671, Apr. 2023.
[27] X. Wei, L. Cai, N. Wei, P. Zou, J. Zhang, and S. Subramaniam, "Joint UAV trajectory planning, DAG task scheduling, and service function deployment based on DRL in UAV-empowered edge computing," *IEEE Internet Things J.*, vol. 10, no. 14, pp. 12826–12838, Jul. 023.
[28] X. Wang, M. C. Gursoy, T. Erpek, and Y. E. Sagduyu, "Learning-based UAV path planning for data collection with integrated collision avoidance," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 16663–16676, Sep. 2022.
[29] M. Samir, C. Assi, S. Sharafeddine, D. Ebrahimi, and A. Ghrayeb, "Age of information aware trajectory planning of UAVs in intelligent transportation systems: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 12382–12395, Nov. 2020.
[30] Y. Wang et al., "Trajectory design for UAV-based Internet of Things data collection: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3899–3912, Mar. 2022.
[31] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "Multi-UAV path planning for wireless data harvesting with deep reinforcement learning," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 1171–1187, 2021.
[32] H. Chang, Y. Chen, B. Zhang, and D. Doermann, "Multi-UAV mobile edge computing and path planning platform based on reinforcement learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 3, pp. 489–498, Jun. 2022.

Meng Xi (Member, IEEE) received the B.S. and Ph.D. degrees in communication and information engineering from Tianjin University, Tianjin, China, in 2018 and 2023, respectively.

She is currently a Postdoctoral Fellow with Tianjin University. Her research interests include machine learning, deep reinforcement learning, and intelligent unmanned system.

Huiao Dai received the B.S. degree in communication and information engineering from Tianjin University, Tianjin, China, in 2022, where he is currently pursuing the M.S. degree with the School of Electrical and Information Engineering.

He research interest includes deep reinforcement learning and pattern recognition.

Jingyi He received the M.S. degree in electronic and communication engineering from Tianjin University, Tianjin, China, in 2020, where he is currently pursuing the Ph.D. degree.

His research interests include marine information processing, intelligent transportation system, AUV path planning, and AUV cluster.

Wenjie Li received the B.S. degree in the communication and information engineering from Tianjin University, Tianjin, China, in 2023, where he is currently pursuing the M.S. degree with the School of Electrical and Information Engineering.

His research interests include deep reinforcement learning and pattern recognition.

Jiabao Wen (Member, IEEE) received the Ph.D. degree from the School of Electrical and Information Engineering, Tianjin University, Tianjin, China, in 2021.

He is currently an Associate Researcher with Tianjin University. His current research interests include data processing, cloud computing, image quality assessment, and Internet of Things.

Shuai Xiao (Member, IEEE) received the Ph.D. degree from the School of Electrical and Information Engineering, Tianjin University, Tianjin, China, in 2022.

He is currently an Associate Researcher with Tianjin University. His research interests are image processing, image quality assessment, computer vision, and image forensics.

Jiachen Yang (Senior Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in communication and information engineering from Tianjin University, Tianjin, China, in 2002, 2005, and 2009, respectively.

He is currently a Professor with the School of Electronical and Information Engineering, Tianjin University. From 2014 to 2015, he was a Visiting Scholar with the Department of Computer Science, School of Science, Loughborough University, Loughborough, U.K. In 2019, he was a Visiting Scholar with Embry-Riddle Aeronautical University, Daytona Beach, FL, USA. He is the Leader of the Laboratory of Stereo Visual Information Processing, Tianjin University. His research interests include image processing, artificial intelligence, and information security. In these areas, he has published more than 100 technical articles in refereed journals and proceedings, including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, and IEEE TRANSACTIONS ON BROADCASTING.

Prof. Yang is on the editorial boards of IEEE ACCESS, *Sensors*, and *Multimedia Tools and Applications* and held special issue on IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE ACCESS, and *Sensors* as a Lead Guest Editor.