IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 December 2023, accepted 14 December 2023, date of current version 5 January 2024.

Digital Object Identifier 10.1109/ACCESS.2023.3346917

RESEARCH ARTICLE

Sensing-Aware Deep Reinforcement Learning With HCI-Based Human-in-the-Loop Feedback for Autonomous Nonlinear Drone Mobility Control

HYUNSOO LEE AND SOOHYUN PARK
Department of Electrical and Computer Engineering, Korea University, Seoul 02841, Republic of Korea

Corresponding author: Soohyun Park (soohyun828@korea.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF-Korea) under Grant 2021R1A4A1030775, and also in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (Ministry of Science and ICT (Information and Communications Technology)) (2022-0-00907, Development of Artificial Intelligence (AI) Bots Collaboration Platform and Self-Organizing AI).

ABSTRACT This paper presents a novel approach for enhancing autonomous drone mobility control using deep reinforcement learning (DRL), primarily aimed at improving autonomous navigation in challenging environments. Our research tackles the significant issue of real-time obstacle avoidance, a critical aspect in drone control. This is achieved through the integration of sensing-aware nonlinear control mechanisms, facilitating advanced trajectory optimization. A notable contribution of our work is the incorporation of real-time human-in-the-loop feedback through human-computer interaction (HCI), which is crucial when pre-trained DRL models encounter environments they are not fully adapted to. Combining autonomous DRL control and HCI feedback equips our system with the flexibility to handle unforeseen scenarios effectively. Furthermore, the paper showcases a comprehensive software demonstration employing Unity 3D for visualization. This demonstration highlights the practical application of our sensing-aware nonlinear control and the HCI-based feedback system, using keyboard interfaces for real-time interaction. The accompanying demonstration video distinctly exhibits the ability of our proposed algorithm.

INDEX TERMS Autonomous mobility control, drone, reinforcement learning, unity, HCI, human-in-the-loop.

I. INTRODUCTION

Unmanned aerial vehicles (UAVs), commonly referred to as drones, possess a wide array of applications across numerous industries. Due to recent advancements in batteries and other complementary technologies, drones have become increasingly adaptable and competent, expanding their roles in surveillance, delivery, and passenger transport functions. Specifically, drones are progressively employed for surveillance purposes, particularly in security and smart city operations [1]. These devices offer aerial perspectives over extensive areas, capturing high-resolution images and videos valuable for situational awareness, monitoring crowd behaviors, and identifying potential threats. By utilizing advanced sensors and machine learning algorithms, drones can undertake more complex surveillance tasks, such as pinpointing and tracking specific individuals or vehicles. Drones have found application in the delivery of goods, especially in areas that are challenging to access or where conventional transportation methods prove unfeasible. These devices can transport a variety of goods, encompassing medical supplies, emergency equipment, and consumer goods [2]. Delivery drones are particularly beneficial in disaster or war zones, where traditional means of transportation may be disrupted or unavailable. Furthermore, drones have recently been investigated as a potential means of passenger

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Quan.

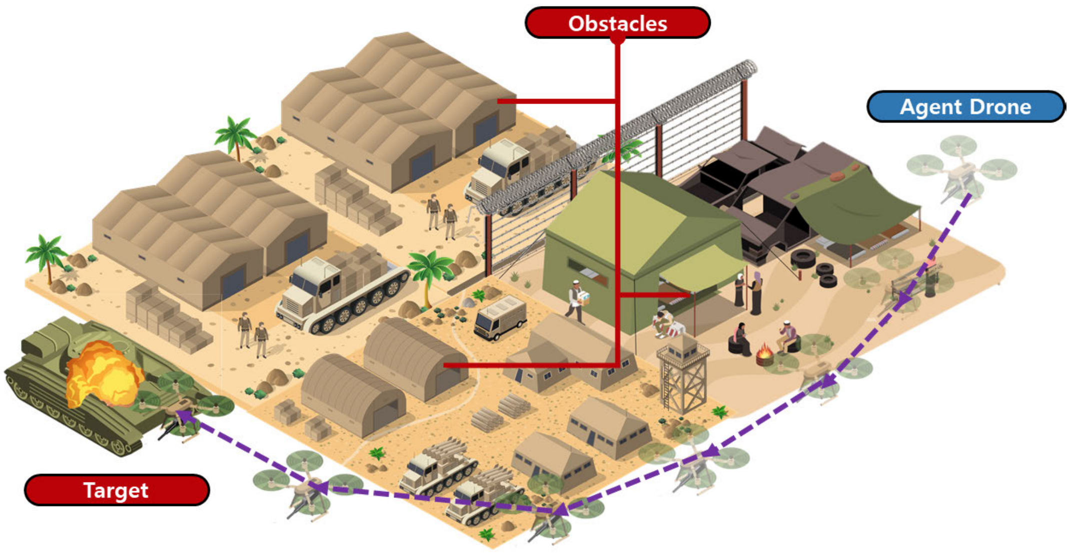

FIGURE 1. Scenario that the drone avoiding obstacles and reach to the target.

transport. Passenger drones present several advantages, including extended operation time as well as decreased traffic congestion [3]. Nevertheless, substantial technical and regulatory obstacles must be overcome before passenger drones can emerge as a viable transportation option [4].

Drones designed for military purposes have emerged as essential tools in contemporary warfare, offering capabilities such as surveillance, military supply delivery, and targeted strikes [5], [6]. These drones possess the advantage of operating in environments that may be too hazardous or inaccessible for human pilots. Nonetheless, employing military drones effectively necessitates the use of advanced technologies and sophisticated algorithms to ensure their safety and performance. One promising approach involves applying machine learning techniques, particularly reinforcement learning (RL), to enhance the autonomy and decision-making abilities of these systems. RL allows military drones to learn from their experiences and adapt their behavior to suit diverse environments and mission requirements.

In real-world applications of DRL-based autonomous drone mobility control, drones may encounter obstacles such as buildings or structures while en route to their target destinations. Consequently, it becomes vital to develop autonomous drone mobility control that incorporates near-field situation sensing for obstacle avoidance. As such, autonomous mobility control algorithms are devised and executed using a range of methods to meet these requirements.

A promising approach for designing and implementing algorithms for autonomous drone control across various use-cases involves employing deep reinforcement learning (DRL) methodologies. These methodologies constitute a stochastic decision-making control process aimed at maximizing expected utility [7], [8]. Yun et al. conducted a study where they jointly optimized the location of drone taxis and passengers' destinations using reinforcement learning based on G2ANet [7]. Also, Tarekegn et al. achieved reasonable performance in terms of communication coverage and network throughput by optimizing the air-to-ground channel using deep Q-learning when deploying drone base stations for wireless communication services [8]. Unlike [7] and [8], the proposed method in this paper does not optimize the overall trajectory but rather deals with the drone's movement as an action, making it more detailed in terms of control. The application of DRL methods can also enhance the performance of military-purpose drones in tasks such as target detection and tracking while minimizing the risk of accidents and tactical damages. Research on autonomous drone mobility control using DRL [9], [10], [11] has been extensively and actively pursued for a wide range of applications in order to fully leverage the potential of drone-related technologies. These applications include surveillance [12], mobile cellular access services [13], [14], [15], and loitering munition [6], [16].

A. RELATED WORK

RL is a category of machine learning techniques wherein an agent makes decisions to maximize its reward through interaction with its environment. This process involves trial and error, as the agent learns from its experiences and adapts its behavior accordingly. In essence, the agent receives feedback in the form of a reward signal, which it utilizes to update its policy or strategy. Recent advancements, such as DRL and model-based RL, have contributed to substantial improvements in complex decision-making processes for a variety of real-world problems, including robotics [17], [18], game-playing [19], and mobility services [20]. One of the key challenges in controlling moving objects is ensuring that the object is aware of its surroundings. In real-world environments, this can be accomplished with the use of sensors like cameras or light detection and ranging (LiDAR) that supply information about the object's environment [21]. Nevertheless, training an RL agent directly is challenging. Therefore, the RL agent is trained in a simulated environment where it can learn to navigate through various scenarios [17]. Numerous 3D tools are available for the visualization and performance evaluation of RL. MuJoCo, a physics simulation engine, caters to a wide range of robotic environments [22]. This engine offers a flexible and intuitive framework for designing and implementing control algorithms for intricate systems. Gazebo is a 3D simulation environment extensively employed to test and assess a diverse array of robotic systems, including those trained with RL [23]. Gazebo supports integration with various robot middleware frameworks, such as the robot operating system (ROS) and open robot control software (OROCOS), which are beneficial for simulating robots. Unity is a gaming engine capable of modeling the physical real-world environments for simulation purposes. It encompasses innovative tools for creating and manipulating 3D objects and environments, as well as a variety of visualization tools. Notably, Unity has recently been extensively utilized for drone-based simulations [24], [25]. However, while [24], [25] employed Unity for observations via drones, they did not carry out any learning processes. In this research, Unity ML-Agents is used, a Unity plugin that enables RL simulations in 3D environments [26].

B. CONTRIBUTIONS

This paper presents the 3D visualization using Unity for autonomous mobility control with run-time control using sensing-awareness, which has not been studied before, to the best of our knowledge [20], [27], [28], [29]. In [27], humans intervene in the process of training the policy, whereas our proposed method does not involve direct intervention in the policy. Rather, it adds human input when the trained model is making inferences in the simulation environment. Applying human intervention to a drone in flight using a fully trained RL algorithm can significantly enhance its avoidance and attack success rates, as clearly demonstrated through experiments. For algorithm validation through 3D visualization, utilizing the Unity platform is essential. In addition, other related research is confined to specific scenarios, such as UAV base stations, which distinguishes them from the problem we propose. On top of DRL, sensing-aware nonlinear control and real-time HCI-based human-in-the-loop intervention/feedback are essential to deal with unexpected situations and emergency scenarios.

II. HUMAN-IN-THE-LOOP FEEDBACK FOR AUTONOMOUS MOBILITY CONTROL

A. ALGORITHM DESIGN CONCEPTS

Among various DRL-based algorithms for *nonlinear* autonomous drone mobility control, deep deterministic policy gradient (DDPG) is particularly beneficial since it is designed to handle continuous action spaces [30]. Fig. 2 illustrates the reward in relation to the training step of Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO) algorithm, and DDPG, which serves as the base algorithm applied. Two reasons support the choice of DDPG as the base algorithm when comparing these three methods. Firstly, it demonstrates the most stable performance, as evidenced by the unstable performance of SAC and PPO algorithms in Fig. 2. Second, the algorithm operates with a fast running time. Table 1 presents the average number of steps processed by each algorithm per second. While the PPO algorithm's training speed is over three times faster than DDPG, it is crucial to consider the time required to achieve a performance level comparable to DDPG. Furthermore, since DDPG's training speed is triple that of SAC, it is well-suited for learning in rapidly changing battlefield environments. This result is also depicted in Fig. 4.

To address unexpected situations, supplementary methods are required to augment the DDPG-based autonomous drone mobility control. Specifically, these methods include sensing-aware nonlinear control and real-time human-in-the-loop intervention. Sensing-aware nonlinear control aims to provide automated situational awareness during drone flights. For implementation, a Unity-based 3D visualization for sensing-aware nonlinear control is realized [26], and situational/sensing-awareness is considered using built-in

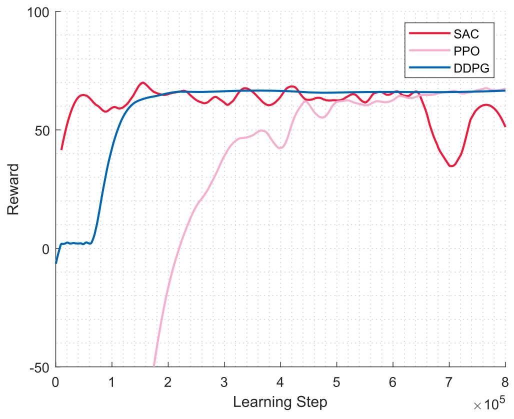

FIGURE 2. Total reward of the algorithm.

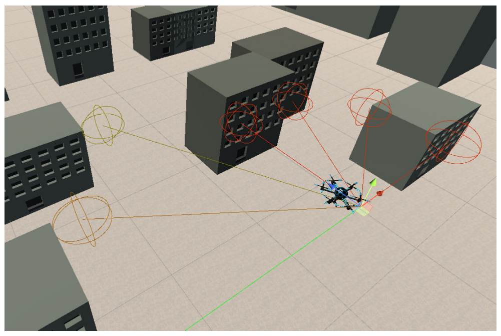

FIGURE 3. Problem scenario: Buildings and Raycast sensing.

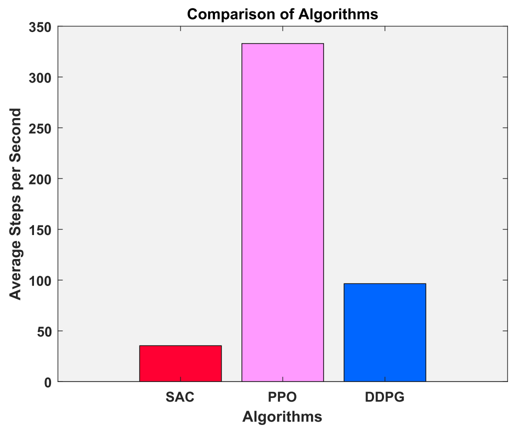

FIGURE 4. **Average learning steps per second.**

TABLE 1. **Comparison of average steps for three algorithms per second.**

Algorithm	Average Steps per Second
SAC	35.4
PPO	332.9
DDPG	96.5

functions. Despite the utilization of sensing-aware nonlinear control, complete automation of mobility control remains challenging due to buildings and obstacles in dense urban environments. Consequently, human-in-the-loop intervention is additionally considered to enable direct drone agent control by users. To achieve this, a Unity-based implementation is conducted to facilitate efficient human-computer interaction (HCI) for controlling the drone agent while in flight in real-time. The roles of humans/users in interacting with drones are actively researched, as they can significantly enhance system flexibility and situational awareness [31]. The proposed algorithm in this paper is fundamentally based on DRL; furthermore, HCI-based human-in-the-loop feedback is added to achieve i) sensing-aware nonlinear control and ii) real-time human-in-the-loop intervention. The DRL-based autonomous drone mobility control is designed and implemented using DDPG, notable for its application in continuous-domain action selection [32]. In the considered drone mobility control, the action is defined based on moving directions (i.e., x-, y-, and z-axes), and the degrees of the three moving directions should be defined within a continuous domain. As a result, the DDPG-based algorithm design is deemed essential. Building upon this foundation, i) sensing-aware nonlinear control and ii) real-time human-in-the-loop interventions are discussed in the following section.

B. ALGORITHM DESIGN CONSIDERATION FACTORS

1) SENSING-AWARE NONLINEAR CONTROL

The considered drone agent has basic state information, which includes i) the distance to the target, ii) the drone's velocity, and iii) the drone's angular velocity. In practical scenarios, the drone gets the information by using built-in range, velocity, and gyro sensors. In simulations, these values are obtained with the Unity ML-Agents built-in function `Sensors.VectorSensor.AddObservation`, which is the function of the sensory input. ML-Agents toolkit enables all communication exchanges between the agent and the simulation environment, facilitating interactions for information sharing. Based on the reward function established in the proposed DDPG-based algorithm, the drone agent determines its autonomous mobility control as follows: i) it flies in a direction where the distance between the drone and the target decreases (based on reward settings); ii) it maneuvers around obstacles to avoid collisions. The detailed reward design for mobility control dynamics is described in Sec. III-A. To detect obstacles, `Raycast` is employed as additional state information. The `Raycast` is a built-in function in Unity that casts rays and obtains distance and direction information of nearby obstacles using the reflection data from the rays, as depicted in Fig. 3. If the rays hit obstacles, the DDPG reward value accumulates additional negative rewards, which are inversely proportional to the distances between the drone and the obstacles (i.e., the absolute additional negative reward increases as the distance becomes closer). Ultimately, it is confirmed that the drone agent flies directly towards the target using the DDPG-based DRL. Moreover, `Raycast` is leveraged for sensing-aware nonlinear control to enable adaptive avoidance of nearby obstacles.

2) REAL-TIME HUMAN-IN-THE-LOOP INTERVENTION

As the considered drone agent navigates towards its target using the proposed DDPG-based DRL in real-time, unexpected situations may arise. For example, avoidance capability may diminish in unanticipated environments with a high density of obstacles. To enhance obstacle avoidance performance, HCI-based human-in-the-loop feedback is integrated with the DDPG-based DRL. For the human-in-the-loop feedback, the first-person shooter concept is employed, where feedback is provided using keyboard directional keys (i.e., up-arrow for moving up vertically, down-arrow for moving down vertically, left-arrow, and right-arrow, respectively) and four pre-defined WASD controls (i.e., W, S, A, and D represent moving forward and backward horizontally, left, and right, respectively). The influence of the human factor reflects 1.25 times more potent than that of the RL algorithm in the simulation to ensure that even if the algorithm's output produces suboptimal behavior, it can be corrected through user intervention. Ultimately, through this HCI-based human-in-the-loop feedback, real-time control of the drone agent's flight movement can be achieved, resulting in successful mobility control.

C. ALGORITHMIC PROCEDURE

1) APPLICATION DOMAIN

The considered application involves a novel autonomous defense system using aerial drones. The drone agent is advantageous in this specific context for several reasons:

- *Easy to Access:* The drone agent can reach its destination or target through aerial routes, making it more accessible compared to ground mobility platforms [33], [34].
- *Easy to Control:* The drone agent enable effortless control using various DRL algorithms [20], [35], [36], enabling easy management of 3D aerial movement.
- *Easy to Deploy:* The drone agent can be deployed in various environments due to its cost-effectiveness. If drone agents were expensive, they would not be disposable. However, drone agents are actively used for loitering munition in modern battlefields because the drone platforms are affordable [37].

Utilizing this drone agent, which offers easy access, control, and deployment, a novel autonomous defense system is designed to employ disposable drone agents for the destruction of incoming enemy tanks. In addition to DRL-based training and inference, mission-specific functionalities for recognizing and avoiding obstacles in urban building regions are essential. When drone agents initiate movement upon detecting enemy tanks, autonomous mobility control techniques are crucial since humans cannot constantly monitor the battlefields. Following this initial movement of drone agents, additional adjustments are necessary to address unexpected situations, such as unpredictable nonlinear movement changes of enemies or unexpected environmental alterations. To this end, the proposed algorithm introduces *i)* sensing-aware nonlinear control and *ii)* human-in-the-loop intervention using HCI-based feedback.

2) PROBLEM SCENARIO

Fig. 3 illustrates the urban environment with high-density buildings. The entire demonstration setup and organization are implemented using Unity. Through Unity-based implementation, numerous buildings can be arranged with 3D visualization. Additionally, the function called `Raycast` is employed for sensing-aware nonlinear mobility control.

3) DRL ALGORITHM ARCHITECTURE AND TRAINING DESCRIPTIONS

The DDPG algorithm structure is depicted in Fig. 5. The actor network serves as the policy function in the proposed DDPG-based framework. Its primary role is to map the observed states to continuous actions, enabling the drone agent to interact effectively with the environment. The architecture of the actor network is a feed-forward neural network comprising three layers. The input has a size corresponding to the dimensionality of the state space. The network includes two hidden layers, each consisting of 128 neurons. The activation function used for these hidden layers is the rectified linear unit (ReLU). The output layer utilizes a hyperbolic tangent (tanh) activation function, ensuring that the output action values are normalized between -1 and 1.

In contrast, the critic network approximates the state-action value function. The input accepts a state vector and passes it through a hidden layer comprising 128 neurons with ReLU activation. Following this, the network concatenates the action vector $\mathbf{a}$ to the resultant state embeddings before forwarding them through the second hidden layer, which also consists of 128 neurons. The final layer of the critic network outputs a single real-valued number, representing the estimated $Q(\mathbf{s}, \mathbf{a})$ value without any activation function.

D. ALGORITHM PSEUDOCODE

The pseudocode for the proposed algorithm is presented in Algorithm 1. Firstly, the weights θ and ϕ of the actor/critic networks, along with the weights θ' and ϕ' of the target network, are initialized. Subsequently, a replay buffer is initialized at the end of each episode. For each minibatch, φ states are generated randomly, and an appropriate set of actions $\mathbf{a} \in \mathbb{A}$ is assigned for each state $\mathbf{s}$. The drone agent utilizes the `Raycast` observation method to detect obstacles in its surroundings for situation-aware DRL computation, storing this information in the state space $\mathbf{s}$ (line 6). To handle exploration in a continuous action space, an Ornstein-Uhlenbeck (OU) noise $\mathcal{N}$ is applied to the selected action $\mathbf{a}$. OU noise is a typical method for DDPG, which is commonly used in exploration strategies for reinforcement learning. Unlike white noise, which is completely random, OU noise exhibits temporal correlation. This means its current value is influenced by its previous values, giving it a degree of smoothness over time. Next, the state-action pairs $(\mathbf{s}, \mathbf{a})$ are input into the pre-designed drone environments, obtaining the corresponding reward set $\mathcal{R}$ for each pair and observing the next state $\mathbf{s}'$. The actor network takes the state received from the drone as input and generates an appropriate action as output through two fully connected layers. The drone agent then executes the corresponding action $\mathbf{a}$ in the environment and acquires the next state $\mathbf{s}'$. The observed transition pairs from the drone agent are stored as a minibatch $\mathcal{V}$.

In phase 2, the actor and target networks are updated. If the timestep falls within the update period, a random minibatch $\mathcal{V}$ is drawn without replacement from the replay buffer $\mathcal{D}$ at each time step. In the policy decision process, learning can be conducted on samples within the replay buffer to mitigate the correlation issue between samples, thereby enhancing the learning outcomes. The equation for computing the network's temporal difference (TD) target is provided in (line 16) of Algorithm 1. The randomly sampled tuple delivered to the replay buffer then serves as input for the networks. Ultimately, this algorithm updates the target critic and target actor networks during $\mathcal{T}$ timesteps and concludes the computation procedure.

Utilizing target networks can improve learning stability. The process sequence of updating the critic network and the two target networks through the loss function and the policy gradient method aims to maximize the expected reward, thereby generating more suitable actions for the actor network.

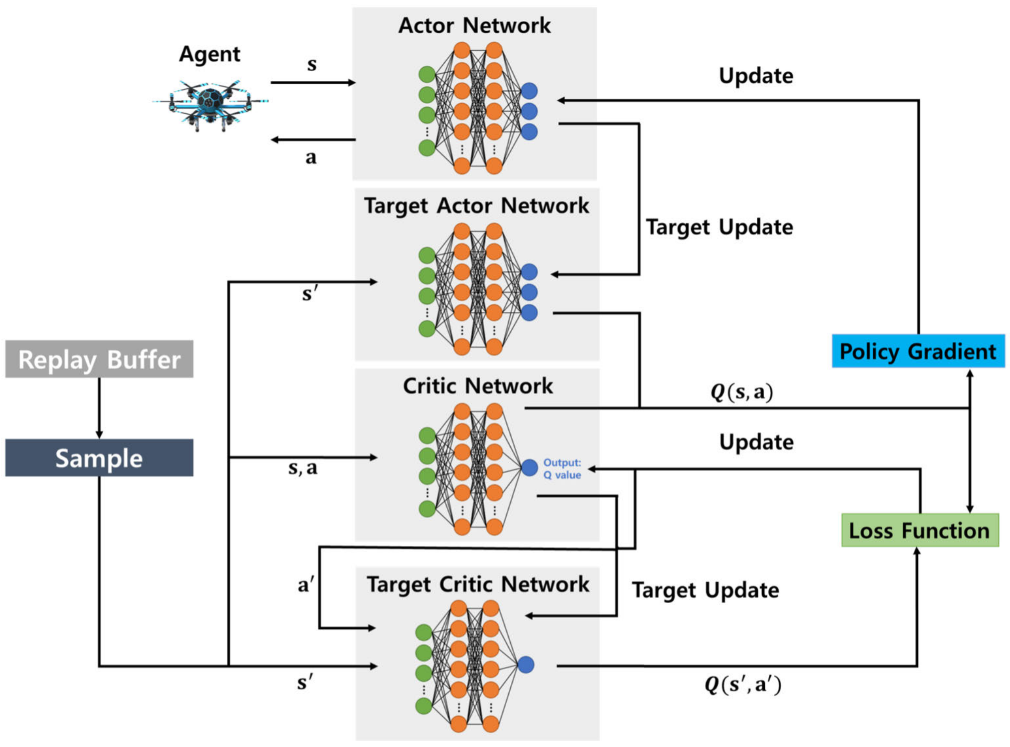

FIGURE 5. DDPG algorithm structure.

TABLE 2. Hyperparameters of the proposed model.

Parameter	Value
Learning step	100,000
Minibatch size	128
Replay buffer size	10,000
Discount factor	0.9
Number of Fully-Connected Layers	2
Learning rate of critic networks	5e-4
Learning rate of actor networks	1e-4
Optimizer	AdamOptimizer
Activation function of hidden layer	ReLU
Activation function of output layer (actor)	tanh

III. PERFORMANCE EVALUATION

A. SETUP AND EVALUATION SETTINGS

In the application software, the following versions of the programs and APIs were utilized: Unity ver. 2021.2, Python ver. 3.9, PyTorch ver. 1.12.1, and Unity ML-Agents ver. 2.0.1 along with their respective packages. Although it is possible to work with other versions, using different versions might lead to errors or compatibility issues.

A city environment that replicates an urban military battlefield was created using the Unity Asset Store. Drone and tank assets were added to the environment scene, with the drone serving as an agent and the tank-shaped target as a game object. The learning computation was executed for 100,000 steps, followed by 10,000 test steps. The batch size for each neural network was set to 128, and a discount factor of 0.9 was applied for each step. The size of the replay buffer was set to 10,000, respectfully. The critic network aimed to reduce the difference between predicted and target values by updating the Q-function. Meanwhile, the policy of the actor network was updated through two fully connected layers to maximize the objective function. The learning rate of the critic network was set to $5e$-4, while for the actor network, it was set to $1e$-4. The hyperparameters of the proposed model are detailed in Table 2. Agents and targets were generated randomly within a designated area, and the target moved forward at a constant speed. To discourage the agent from remaining stationary, a continuous default reward of -0.01 was applied. A compensation function was incorporated to encourage the drone to approach the target gradually, reflecting the differences in distances. The drone received a reward of $+1$ upon reaching the target and a reward of -1 for being too far away or colliding with an obstacle, which also ended the episode. A negative reward was provided when the drone identified a nearby obstacle using `Raycast` (i.e., situation-awareness) to promote learning of obstacle avoidance. The proposed situation-aware autonomous nonlinear drone mobility control algorithm outperformed conventional DDPG-based DRL algorithms by utilizing information about surrounding obstacles. For human-in-the-loop feedback, the force exerted on the agent through keyboard input was configured to be 1.25 times the force with which the drone moves autonomously.

B. EVALUATION RESULTS

We aimed to create an environment as close to reality as possible using Unity. 3D visualization is not only necessary

Algorithm 1 Proposed Sensing-Aware DRL With Human-in-the-Loop Feedback

```
1:  Initialize the critic and actor networks
2:  Initialize the target networks as: $\theta' \leftarrow \theta, \phi' \leftarrow \phi$
3:  for episode = 1, $\mathcal{E}$ do
4:      Phase 1. Initialize the replay buffer $\mathcal{D}$
5:      ▹ Randomly generate $\varphi$ states $\mathbf{s} \in \mathbb{S}$
6:      ▹ Observe surrounding obstacles through Raycast function and store to the states $\mathbf{s}$ to aware situation
7:      ▹ Get corresponding set of actions with OU noise $\mathbf{a} = \mu(\mathbf{s}|\phi') + \mathcal{N}$, $\mathbf{a} \in \mathbb{A}$ for each $\mathbf{s}$
8:      ▹ Input the state-action pairs to predefined drone environments and get a set of reward $\mathcal{R}$ for each pair, and observe the next set of states $\mathbf{s}' \in \mathbb{S}$
9:      ▹ Store the transition pairs $(\mathbf{s}, \mathbf{a}, \mathcal{R}, \mathbf{s}')$ as a minibatch, which composes $\mathcal{D}$.
10: end for
11: Phase 2. Update neural networks periodically
12: for time step = 1, $\mathcal{T}$ do
13:     If time step is update period, do followings:
14:         ▹ Sample a random minibatch $\mathcal{V} = (\mathbf{s}^j, \mathbf{a}^j, \mathcal{R}^j, \mathbf{s}'^j)$ without replacing from $\mathcal{D}$
15:         ▹ Set $y_i = \mathcal{R}_i^j + \gamma Q^{\mu'}(\mathbf{s}_i'^j, \mu(\mathbf{s}_i'|\phi')|\theta')$
16:         ▹ Update the $\theta$ by applying stochastic gradient descent to the loss function of critic network, which can be obtained as $\frac{1}{\mathcal{V}} \sum_i (y_i - Q_i^{\mu}(\mathbf{s}_i^{\prime j}, \mathbf{a}_i^j|\theta))^2$
17:         ▹ Update the $\phi$ by applying stochastic gradient ascent concerning the gradient of actor network:
18:         $\nabla_\theta \mathcal{J} \approx \frac{1}{\varphi} \sum_i \nabla_\theta Q(\mathbf{s}^j, \mathbf{a}^j|\theta) \nabla_\phi \mu(\mathbf{s}|\phi)|_{\mathbf{s}=\mathbf{s}_i, \mathbf{a}=\mu(\mathbf{a}_i|\phi)}$
19:         ▹ Soft update $\theta'$ and $\phi'$ as follows:
20:         $\theta' \leftarrow \gamma\theta + (1-\gamma)\theta', \phi' \leftarrow \gamma\phi + (1-\gamma)\phi'$
21: end for
22: Phase 3. Human-in-the-loop feedback
23: If the performance is not satisfactory during run-time operations, the experimental results can be adjusted through keyboard manipulation as human-in-the-loop intervention.
```

for assessing the algorithm's performance but also crucial for human-in-the-loop feedback. Additionally, the actual flight route of the drone can be more clearly represented in a 3D environment than in 2D. Fig. 6 displays the step-by-step flying movement procedure of the proposed algorithm. From Fig. 6(a) to Fig. 6(h), the algorithm demonstrates efficient nonlinear mobility control that can avoid obstacles and buildings.

Fig. 7 presents the numerical results of successful attacks depending on obstacle density, where the obstacle density is defined as the level of difficulty of the map. At 0% obstacle density, there are no obstructions. For every increment of 10%, a building is randomly placed in the space between the drone agent and the target, culminating in a total of 10 buildings at 100% obstacle density. A high success rate in environments with high obstacle density represents the

TABLE 3. The number of successful attacks per 20 trials based on obstacle density through nonlinear drone mobility control.

Nonlinear DDPG	20%	40%	60%	80%	100%
Round 1	16/20	14/20	10/20	10/20	7/20
Round 2	17/20	15/20	7/20	8/20	7/20
Round 3	13/20	17/20	8/20	11/20	5/20
Round 4	18/20	16/20	9/20	7/20	7/20
Round 5	17/20	16/20	14/20	5/20	10/20
Round 6	17/20	15/20	6/20	11/20	8/20
Round 7	17/20	16/20	14/20	6/20	10/20
Round 8	19/20	17/20	8/20	13/20	4/20
Round 9	15/20	16/20	12/20	7/20	5/20
Round 10	17/20	10/20	11/20	8/20	10/20
Maximum	19	17	14	13	10
Average	16.6	15.2	9.9	8.6	7.3
Median	17	16	9.5	8	7
Minimum	13	10	6	5	4

TABLE 4. The number of successful attacks per 20 trials based on obstacle density through linear drone mobility control.

Linear DDPG	20%	40%	60%	80%	100%
Round 1	17/20	9/20	5/20	4/20	2/20
Round 2	13/20	10/20	4/20	2/20	1/20
Round 3	15/20	6/20	4/20	2/20	1/20
Round 4	14/20	11/20	2/20	3/20	2/20
Round 5	18/20	12/20	9/20	1/20	1/20
Round 6	13/20	7/20	10/20	4/20	0/20
Round 7	11/20	8/20	5/20	3/20	2/20
Round 8	15/20	6/20	5/20	2/20	3/20
Round 9	10/20	4/20	4/20	2/20	0/20
Round 10	14/20	13/20	7/20	4/20	4/20
Maximum	18	13	10	4	4
Average	14	8.6	5.5	2.7	1.6
Median	14	8.5	5	2.5	1.5
Minimum	10	4	2	1	0

robustness of the considering algorithm. In the simulation environment, each experiment was repeated for 10 rounds and 20 times in each environment, estimating the number of successful agent target reaches or destructions in each round. Table 3 and Table 4 show the results of each round in Fig. 7 in detail. The agent successfully reaches the target in all trials if no obstacles are present. Nonetheless, despite the application of a well-trained algorithm, the success rate diminishes when the density of obstacles increases, occasionally necessitating passage through extremely narrow spaces. In the experiments where human feedback was added to the existing algorithm, it was observed that the number of successful trials increased, even when the obstacle density was very high, as depicted in Fig. 7. This finding suggests that the incorporation of human-in-the-loop intervention enhances the performance of the autonomous drone mobility control algorithm, particularly in complex environments with high obstacle densities. Human intervention helps the drone navigate challenging scenarios

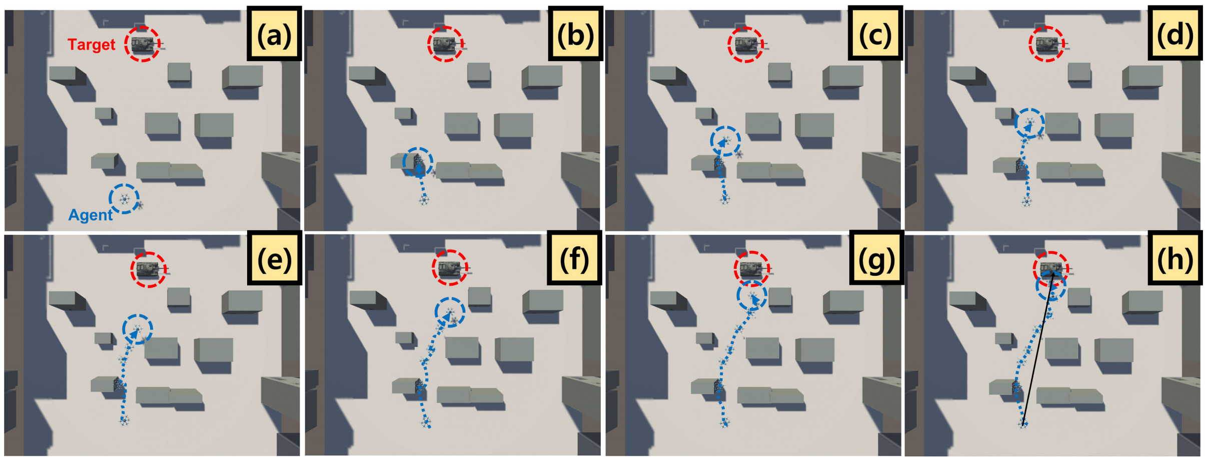

FIGURE 6. Top-down view of drone agent flying movement.

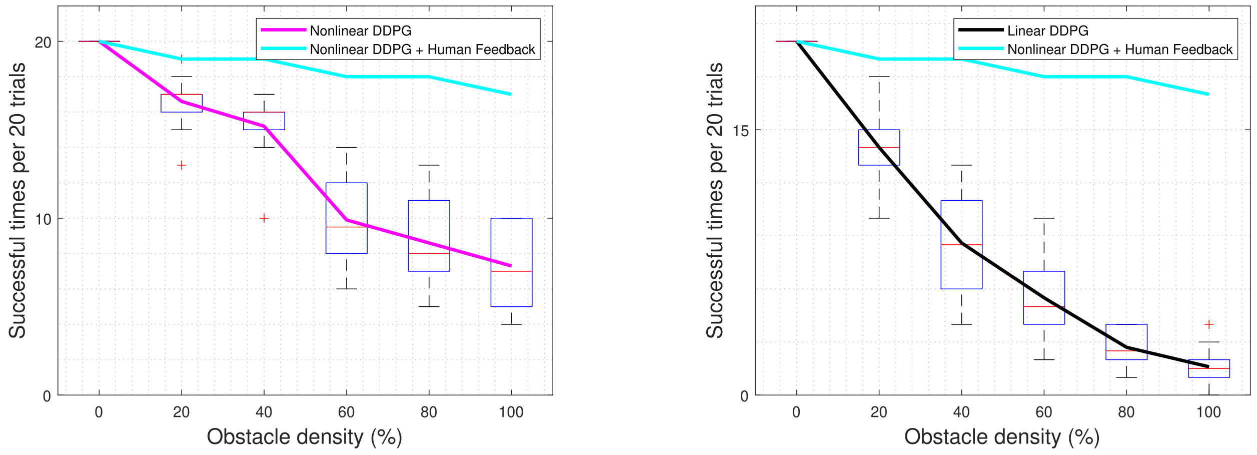

FIGURE 7. The number of successful attacks per 20 trials according to obstacle density.

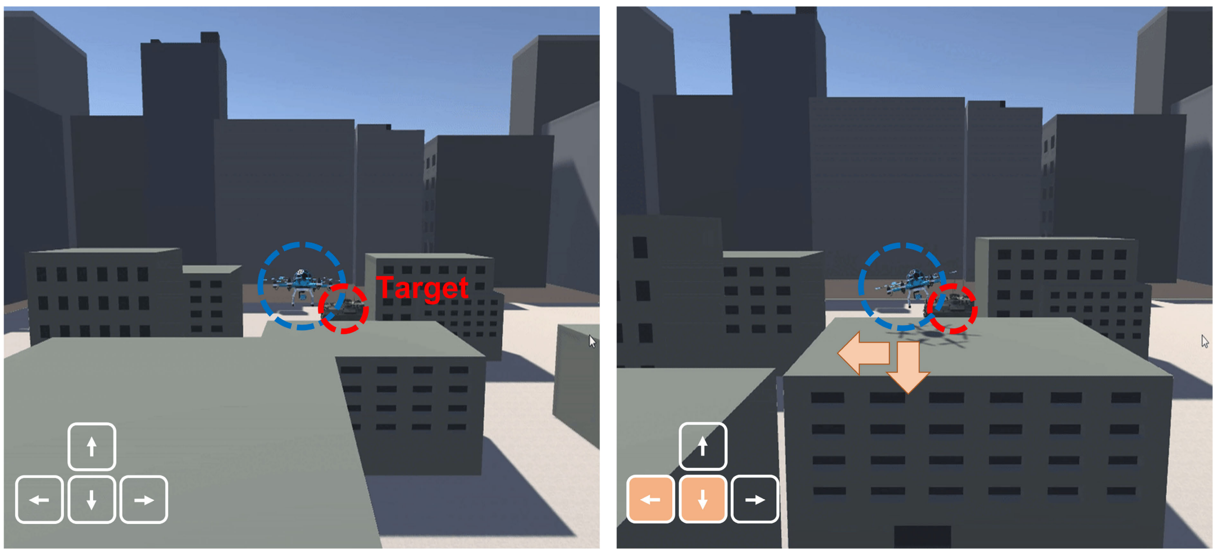

FIGURE 8. Human-in-the-loop feedback for nonlinear mobility control.

and avoid obstacles more effectively, leading to a higher success rate in reaching the target. Fig. 8 illustrates the human-in-the-loop feedback for nonlinear mobility control of the agent. In a well-trained environment, the agent may struggle to reach the goal if there are too many obstacles, as demonstrated in Fig. 6. Human intervention is applied to the trained model to enhance the success rate in reaching the target. When the agent is about to collide with an obstacle or the ground, the user can help the agent avoid the obstacle by moving the drone up, down, left, or right using the direction key input. However, the probability of success may vary significantly depending on the operator's proficiency, so the operator may need extensive training in the simulation environment before applying the technique in the real world. Lastly, the demonstration video of the proposed algorithm is available at https://youtu.be/GrZ_HMX1xms.

IV. CONCLUDING REMARKS

This paper presents the design and implementation of novel autonomous drone mobility control based on DDPG-grounded DRL algorithms. Two additional approaches are considered to achieve more realistic and real-time drone trajectory control, i.e., (i) sensing-aware nonlinear control and (ii) HCI-based real-time human-in-the-loop intervention/feedback. Based on the implementation using Unity ML-Agents, sensing-aware control is implemented through the built-in `Raycast` function, which is an essential part of autonomous nonlinear control. In addition, HCI-based human-in-the-loop feedback is also introduced for real-time adjustment after executing the model in 3D environment. Our experimental results and demonstrations confirm that the proposed sensing-aware autonomous nonlinear mobility control algorithm, combined with HCI-based human-in-the-loop feedback, performs as expected, enhancing drone mobility control in complex environments.

Although this paper focuses on a single drone agent attacking a single target, the potential exists for further scalability. This is due to the opportunity for operating a greater number of drone agents or encountering multiple targets in real-world scenarios. Regarding adaptability, the same algorithm can be sufficiently applied to different scenarios. For example, when operated by the military, this framework appears readily applicable for tasks involving transporting supplies while avoiding obstacles.

ACKNOWLEDGMENT

The authors would like to thank to Joongheon Kim for his advice and contribution on research initiation and discussions.

REFERENCES

[1] A. Gohari, A. B. Ahmad, R. B. A. Rahim, A. S. M. Supa'at, S. A. Razak, and M. S. M. Gismalla, "Involvement of surveillance drones in smart cities: A systematic review," *IEEE Access*, vol. 10, pp. 56611–56628, 2022.

[2] F. Kong, J. Li, B. Jiang, H. Wang, and H. Song, "Trajectory optimization for drone logistics delivery via attention-based pointer network," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4519–4531, Apr. 2023.

[3] W. J. Yun, S. Jung, J. Kim, and J.-H. Kim, "Distributed deep reinforcement learning for autonomous aerial eVTOL mobility in drone taxi applications," *ICT Exp.*, vol. 7, no. 1, pp. 1–4, Mar. 2021.

[4] R. K. Lee, C. A. Kitts, M. A. Neumann, and R. T. Mcdonald, "Multiple UAV adaptive navigation for three-dimensional scalar fields," *IEEE Access*, vol. 9, pp. 122626–122654, 2021.

[5] B. Bera, A. K. Das, S. Garg, M. J. Piran, and M. S. Hossain, "Access control protocol for battlefield surveillance in drone-assisted IoT environment," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2708–2721, Feb. 2022.

[6] D. Hambling, "Failure or savior? Busting myths about switchblade loitering munitions in Ukraine," *Forbes*, Jun. 2022.

[7] W. J. Yun, Y. J. Ha, S. Jung, and J. Kim, "Autonomous aerial mobility learning for drone-taxi flight control," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2021, pp. 329–332.

[8] G. B. Tarekegn, R.-T. Juang, H.-P. Lin, Y. Y. Munaye, L.-C. Wang, and M. A. Bitew, "Deep-reinforcement-learning-based drone base station deployment for wireless communication services," *IEEE Internet Things J.*, vol. 9, no. 21, pp. 21899–21915, Nov. 2022.

[9] D. Shumeye Lakew, U. Sa'ad, N.-N. Dao, W. Na, and S. Cho, "Routing in flying ad hoc networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1071–1120, 2nd Quart., 2020.

[10] D. S. Lakew, W. Na, N.-N. Dao, and S. Cho, "Aerial energy orchestration for heterogeneous UAV-assisted wireless communications," *IEEE Syst. J.*, vol. 16, no. 2, pp. 2483–2494, Jun. 2022.

[11] H. Menouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 22–28, Mar. 2017.

[12] R. Shakeri, M. A. Al-Garadi, A. Badawy, A. Mohamed, T. Khattab, A. K. Al-Ali, K. A. Harras, and M. Guizani, "Design challenges of multi-UAV systems in cyber-physical applications: A comprehensive survey and future directions," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3340–3385, 4th Quart., 2019.

[13] C. Park, H. Lee, W. J. Yun, S. Jung, C. Cordeiro, and J. Kim, "Cooperative multi-agent deep reinforcement learning for reliable and energy-efficient mobile access via multi-UAV control," 2022, *arXiv:2210.00945*.

[14] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1123–1152, 2nd Quart., 2016.

[15] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2334–2360, 3rd Quart., 2019.

[16] H. Lee, S. Park, W. J. Yun, S. Jung, and J. Kim, "Situation-aware deep reinforcement learning for autonomous nonlinear mobility control in cyber-physical loitering munition systems," 2023, *arXiv:2301.00124*.

[17] W. J. Yun, D. Mohaisen, S. Jung, J.-K. Kim, and J. Kim, "Hierarchical reinforcement learning using Gaussian random trajectory generation in autonomous furniture assembly," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manag.* New York, NY, USA: Association for Computing Machinery, Oct. 2022, pp. 3624–3633.

[18] Y. Jia and S. Ma, "A coach-based Bayesian reinforcement learning method for snake robot control," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2319–2326, Apr. 2021.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[20] W. J. Yun, S. Park, J. Kim, M. Shin, S. Jung, D. A. Mohaisen, and J.-H. Kim, "Cooperative multiagent deep reinforcement learning for reliable surveillance via autonomous multi-UAV control," *IEEE Trans. Ind. Informat.*, vol. 18, no. 10, pp. 7086–7096, Oct. 2022.

[21] Y. Han, I. H. Zhan, W. Zhao, J. Pan, Z. Zhang, Y. Wang, and Y.-J. Liu, "Deep reinforcement learning for robot collision avoidance with self-state-attention and sensor fusion," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 6886–6893, Jul. 2022.

[22] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vilamoura-Algarve, Portugal, Oct. 2012, pp. 5026–5033.

[23] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sendai, Japan, Sep. 2004, pp. 2149–2154.

[24] G. Kalitsios, V. Mygdalis, and I. Pitas, "Enhancing power line segmentation for UAV inspection utilizing synthetic data," in *Proc. Int. Conf. Robot. Autom. (ICRA) Workshops The Role Robot. Simulators Unmanned Aerial Vehicles*, London, U.K., Jun. 2023, pp. 1–4.

[25] K. Vivaldini and T. Pazelli, "Unity3D and robot operating system applied to the development of a multi-UAV strategy to identify oil spills around offshore platforms," in *Proc. Int. Conf. Robot. Autom. (ICRA) Workshops Role Robot. Simulators Unmanned Aerial Vehicles*, London, U.K., Jun. 2023, pp. 1–4.

[26] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," 2020, *arXiv:1809.02627*.

[27] C. Celemin and J. Ruiz-Del-Solar, "An interactive framework for learning continuous actions policies based on corrective feedback," *J. Intell. Robotic Syst.*, vol. 95, no. 1, pp. 77–97, Jul. 2019.

[28] A. B. Bhandarkar, S. K. Jayaweera, and S. A. Lane, "Adversarial Sybil attacks against deep RL based drone trajectory planning," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Rockville, MD, USA, Nov. 2022, pp. 1–6.

[29] S. Choudhury, K. Solovey, M. Kochenderfer, and M. Pavone, "Coordinated multi-agent pathfinding for drones and trucks over road networks," in *Proc. ACM Int. Conf. Auto. Agents MultiAgent Syst. (AAMAS)*, 2022, pp. 272–280.

[30] H. Mao, Z. Zhang, Z. Xiao, and Z. Gong, "Modelling the dynamic joint policy of teammates with attention multi-agent DDPG," in *Proc. ACM Int. Conf. Auto. Agents MultiAgent Syst. (AAMAS)*, Montreal QC, Canada, 2019, pp. 1108–1116.

[31] D. Tezza and M. Andujar, "The state-of-the-art of human-drone interaction: A survey," *IEEE Access*, vol. 7, pp. 167438–167454, 2019.

[32] D. Kwon, J. Jeon, S. Park, J. Kim, and S. Cho, "Multiagent DDPG-based deep learning for smart ocean federated learning IoT networks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9895–9903, Oct. 2020.

[33] W. Shi, H. Zhou, J. Li, W. Xu, N. Zhang, and X. Shen, "Drone assisted vehicular networks: Architecture, challenges and opportunities," *IEEE Netw.*, vol. 32, no. 3, pp. 130–137, May 2018.

[34] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.

[35] K. Li, W. Ni, Y. Emami, and F. Dressler, "Data-driven flight control of Internet-of-Drones for sensor data aggregation using multi-agent deep reinforcement learning," *IEEE Wireless Commun.*, vol. 29, no. 4, pp. 18–23, Aug. 2022.

[36] H. Cheng, L. Bertizzolo, S. D'oro, J. Buczek, T. Melodia, and E. S. Bentley, "Learning to fly: A distributed deep reinforcement learning framework for software-defined UAV network control," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 1486–1504, 2021.

[37] T. Zhang, T. Qiu, Z. Liu, Z. Pu, J. Yi, J. Zhu, and R. Hu, "Multi-UAV cooperative short-range combat via attention-based reinforcement learning using individual reward shaping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Kyoto, Japan, Oct. 2022, pp. 13737–13744.

HYUNSOO LEE received the B.S. degree in electronic engineering from Soongsil University, Seoul, Republic of Korea, in 2021. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Korea University, Seoul.

His research interests include deep learning algorithms and their applications to mobility and networking. He was a recipient of the IEEE Vehicular Technology Society (VTS) Seoul Chapter Award, in 2022.

SOOHYUN PARK received the B.S. degree in computer science and engineering from Chung-Ang University, Seoul, Republic of Korea, in 2019, and the Ph.D. degree in electrical and computer engineering from Korea University, Seoul, in 2023.

Since 2023, she has been a Postdoctoral Scholar with the Department of Electrical and Computer Engineering, Korea University. Her research interests include deep learning algorithms and their applications to computer networking, autonomous mobility platforms, and quantum multi-agent distributed autonomous systems. She was a recipient of the IEEE Vehicular Technology Society (VTS) Seoul Chapter Awards, in 2019 and 2023; the IEEE Seoul Section Student Paper Contest Awards, in 2020 and 2023; the HFR Research Paper Award for Quantum Technologies by KICS, in 2023; and the Best Reviewer Award by *ICT Express* (Elsevier), in 2021.