aerospace

MDPI

Article

Robust Guidance Policies Through Deep Reinforcement Learning

Seongyeon Kim [1], Jongho Shin [1,*] and Hyeong-Geun Kim [2]

[1] Department of Mechanical Engineering, Chungbuk National University, Cheongju 28644, Republic of Korea; syk_@cbnu.ac.kr
[2] Department of Aerospace and Mobility Engineering, Konkuk University, Seoul 05029, Republic of Korea; hyeongkim@konkuk.ac.kr
* Correspondence: jshin@cbnu.ac.kr; Tel.: +82-43-261-2447

Abstract

Unmanned aerial vehicle (UAV) guidance systems must operate reliably under significant uncertainties, such as sensor noise, target maneuvers, and environmental disturbances. Traditional guidance methods like proportional navigation (PN), while computationally efficient, often struggle to maintain performance under such challenging conditions. To overcome these limitations, this study proposes a robust UAV guidance framework based on deep reinforcement learning (DRL), specifically utilizing the soft actor–critic (SAC) algorithm. The UAV–target tracking problem is formulated as the Markov decision process (MDP) for both two-dimensional (2D) and three-dimensional (3D) scenarios. A deep neural network policy is trained in noisy environments to generate acceleration commands that minimize the zero-effort miss (ZEM). Extensive numerical simulations conducted using the OpenAI Gym validate effectiveness of the proposed method under previously unseen initial conditions and increased noise levels. The results demonstrate that the SAC-based policy achieves higher tracking success rates than the PN, particularly under strict terminal conditions and observation noise.

Keywords: guidance law; deep reinforcement learning-based guidance law; soft actor–critic; 2D/3D environments; observation noise; numerical simulations

1. Introduction

Unmanned aerial vehicle (UAV) guidance stands as a fundamental technology in the fields of aerospace. Since the inception of guided UAV systems, considerable research efforts have focused on improving both guidance precision and control effectiveness [1]. Among various existing approaches, a proportional navigation (PN) generates acceleration commands for tracking by reducing a line-of-sight (LOS) angular rate between the UAV and the target. Its simplicity and computational efficiency have led to its widespread use in the UAV guidance systems [2]. Despite these advantages, the PN exhibits inherent limitations when applied to scenarios involving nonlinear dynamics, sensor interference, and maneuvering targets. Specifically, as the UAV approaches a target, the LOS angular rate tends to decrease, diminishing the effectiveness of the PN-generated commands. Consequently, this ultimately reduces tracking accuracy [3]. These limitations are even more pronounced in environments where targets may perform evasive maneuvers or where on-board sensors may experience noise and command delays. In such situations, conventional PN methods struggle to ensure robust performance [4].

To overcome the aforementioned limitations, guidance laws utilizing various classical control strategies, such as sliding mode control and optimal control formulation, have been

Check for updates

Academic Editor: Gokhan Inalhan

Received: 15 January 2026
Revised: 11 February 2026
Accepted: 27 February 2026
Published: 2 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

studied [5,6]. These methods are designed to satisfy additional constraints, such as terminal collision angle, field-of-view (FOV) limitations, and time-to-energy efficiency [7]. In [8], non-singular terminal sliding mode guidance was developed to accommodate angular constraints during the terminal phase. In [9], collision time and collision angle control strategies were introduced to improve tracking effectiveness in specific scenarios. These methods have shown superior performance over the PN in structured or deterministic environments. However, their practical implementation remains complex. Many of these control laws rely heavily on accurate system modeling and precise state estimation. Therefore, maintaining their performance in real-world environments is difficult due to sensor noise, disturbances, and uncertain target behavior. Moreover, optimization-based derivation techniques require real-time numerical solvers or iterative computations [10]. This can impose a significant computational burden on systems with limited onboard processing capacity [11].

In recent decades, computational resources and machine learning technologies have rapidly advanced. This has led to a growing interest in data-driven strategies such as deep learning. In this method, neural networks have been used to approximate optimal control policies or reproduce classical guidance laws. In [12], a deep neural network was used as a supervised learning method to mimic the behavior of the PN. In [13], a regression-based prediction model was used to learn to minimize the final failure distance. These approaches have effectively reduced computational load. However, their performance remains limited due to a lack of generalization to unfamiliar conditions during training.

To address these limitations, deep reinforcement learning (DRL) has recently attracted attention as an alternative. Unlike supervised learning, this method promotes autonomous exploration within the action space, allowing the agent to maximize its cumulative reward [14]. This allows policy improvement through numerous learning iterations. These characteristics make it particularly suitable for control problems with high-dimensional state and action spaces, such as missile guidance [15]. Furthermore, DRL is suitable for robust and adaptive guidance because it can dynamically adapt through learning in uncertain and variable environments containing noise [16].

Various DRL algorithms such as the deep deterministic policy gradient (DDPG) [17], proximal policy optimization (PPO) [18], and soft actor–critic (SAC) [19] have been applied to missile guidance problems [20–22]. These methods have been shown to work effectively even in cases with stochastic disturbances or uncertain target behaviors. In [23], it was shown that DRL-based guidance laws achieved lower miss distances compared to the conventional PN in environments with sensor noise. In [24], a kernel-based online RL framework capable of real-time adaptation to maneuvering targets was proposed, and it showed partially improved performance.

However, existing studies have been limited to two-dimensional models [24]. In other words, relatively low effort has been made to address the full complexity of three-dimensional tracking. Moreover, the range of initial condition settings is often limited. This can lead to policy overfitting to a narrow state space and hinders adaptability to changes in the actual environment. Moreover, most training and validation environments are configured on the premise of ideal observation conditions [25]. In other words, it is difficult to secure the robustness of the policy against uncertainties such as sensor noise and observation errors that may occur during actual operation.

In particular, there are many cases where the initial condition setting range is limited [14,26]. This may cause a problem in which the policy is overfitted to a narrow state space, and reduces adaptability to actual environmental changes. In addition, even in the design of the compensation function, there were many cases where the physical criteria for improving the guidance performance were not sufficiently reflected. For example, in many studies, compensation was constructed based on the final distance or simple

weight. In other words, designs that reflect zero-effort miss (ZEM) that can quantitatively represent the actual performance are rare [25,26].

This study aims to address these issues by proposing a DRL-based guidance law in both two-dimensional and three-dimensional environments. To achieve this, SAC algorithm is utilized. The SAC is an off-policy, entropy-regularized, and sample-efficient method known for its performance in continuous control tasks. The UAV–target tracking problem is formulated as a Markov decision process (MDP) [27] for both 2D and 3D tracking scenarios, with explicit inclusion of sensor noise and unseen initial configurations. Unlike prior works limited to 2D cases or narrow state-space generalization, this study constructs a comprehensive training and evaluation framework using OpenAI Gym [28], with models exposed to wide-ranging and randomized conditions. Numerical experiments using the proposed method have verified that the learned policy can effectively respond to various noises and target movements.

The contributions of this study are summarized as follows:

- Noise-contaminated DRL environments for 2D/3D tracking: The UAV–target tracking scenarios in both 2D and 3D are formulated as MDPs while explicitly injecting Gaussian observation noise into LOS and range measurements, enabling robustness to sensor uncertainty to be learned during training.
- Development of robust guidance laws using the SAC: The SAC algorithm is employed to construct continuous action guidance policies that remain stable across varying noise levels, demonstrating robustness even under disturbances 50 times stronger than those used during training.
- Reward function based on zero-effort miss (ZEM): The reward function is constructed directly from the zero-effort miss (ZEM), providing a physically grounded terminal-accuracy metric that improves precision in final tracking phases where the LOS rate diminishes and conventional PN becomes less effective.
- Full 3D MDP and wide initial condition: The guidance problem is extended to full 3D with yaw and pitch LOS dynamics and is trained across a broad range of random initial conditions, including yaw/pitch angles up to ±90°, ensuring generalization across diverse tracking geometries.
- Comprehensive comparison with PN-based guidance: Through extensive simulations, the proposed method is compared with traditional PN under various termination distances and noise levels, showing superior performance, especially in close-range trackings and noisy conditions.

The remainder of this paper is structured as follows: The 2D and 3D kinematic models of the UAV and target are defined in Section 2. In Section 3, the DRL-based guidance law is proposed by defining the UAV follow situation as the MDP. The numerical simulations are conducted with the proposed method and the results are analyzed in Section 4. Finally, Section 5 concludes this paper.

2. Kinematic Model for Tracking

In this section, the 2D and 3D kinematic models are derived to describe the motion of the UAV and target.

2.1. 2D Kinemtic Model

In this subsection, the 2D tracking geometry for target tracking is defined and described. The 2D tracking can be mathematically interpreted as a special case of 3D tracking. In this study, the 2D kinematic model plays a crucial role. The validity of the Markov decision process formulation, reward design, and reinforcement learning framework is verified through a 2D environment. In particular, most classical guidance laws, including

PN, have been derived and analyzed from 2D tracking kinematics. Therefore, 2D tracking enables direct comparison with existing guidance theories and facilitates the physical interpretation of learned policy behavior. Furthermore, dimensionality reduction facilitates stable reward formation and hyperparameter tuning before extending the framework to 3D tracking, where state–action coupling becomes more complex. Figure 1 shows the 2D tracking geometry. The inertial reference frame is denoted by (X_I, Y_I). The variables (X_U, Y_U) represent the body frame of the UAV, and variables (X_T, Y_T) represent the body frame of the target. Each frame is rotated by an angle of (ψ_U, ψ_T) from the inertial reference frame. The 2D kinematics of the UAV and target are as follows:

$$\begin{aligned} \dot{x}_i &= V_i cos\psi_i, \\ \dot{y}_i &= V_i sin\psi_i, \\ \dot{\psi}_i &= \frac{a_{y_i}}{V_i}, \end{aligned} \tag{1}$$

where i denotes the UAV (U) and target (T). Variables x_i, y_i denote the x, y position in the inertial coordinate frame, and variables V_i, ψ_i, a_i indicate velocity, flight path angle, and vertical acceleration, respectively.

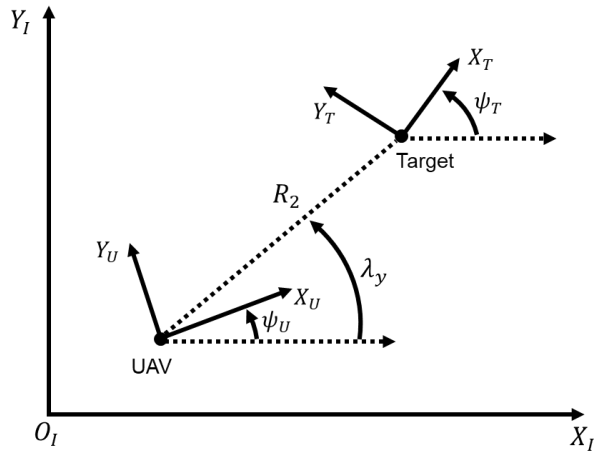


Figure 1. 2D tracking geometry.

The UAV moves in the X_U axial direction with a velocity V_U, and maneuvers laterally with an axial acceleration a_{y_U}. The target moves in the direction of X_T with velocity V_T without axial acceleration. The variable R_2 denotes the relative Euclidean distance between the UAV and the target. The variable λ_y means the angle of LOS in the inertial reference frame. The 2D tracking kinematics of the UAV and target are described as follows:

$$\begin{aligned} \dot{R}_2 &= V_T \cos(\psi_T - \lambda_y) - V_U \cos(\psi_U - \lambda_y), \\ R_2\dot{\lambda}_y &= V_T \sin(\psi_T - \lambda_y) - V_U \sin(\psi_U - \lambda_y). \end{aligned} \tag{2}$$

2.2. 3D Kinematic Model

In this subsection, the 3D tracking geometry is defined and described. Figure 2 shows the 3D tracking geometry. The inertial reference frame is denoted by (X_I, Y_I, Z_I). The variables (X_U, Y_U, Z_U) denote the body-fixed frame of the UAV, while the body-fixed frame of the target is represented with (X_T, Y_T, Z_T). It is assumed that the UAV moves at the velocity of V_U in the direction of the X_U axis, and the motion along the directions of Y_U and Z_U are determined using the acceleration values a_{y_U}, a_{z_U}. The motion of the target is represented in a similar manner to that of the UAV. The acceleration of the target is a

constant, a_{y_T}, and a_{z_T} is applied by 5 g. The 3D tracking kinematics of the UAV and target are described as follows:

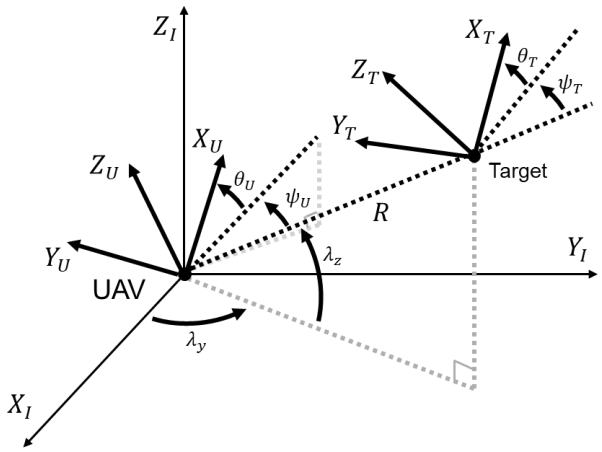


Figure 2. 3D tracking geometry.

$$\begin{aligned} \dot{R} &= V_T \cos\theta_T \cos\psi_T - V_U \cos\theta_U \cos\psi_U, \\ R\dot{\lambda}_y &= V_T \cos\theta_T \sin\psi_T - V_U \cos\theta_U \sin\psi_U, \\ R\dot{\lambda}_z &= -V_T \sin\theta_T + V_U \sin\theta_U, \end{aligned} \tag{3}$$

where $\dot{\lambda}_y$ and $\dot{\lambda}_z$ are the LOS angular rates in the y and z directions in the inertial frame, respectively. The variables $\theta_U, \psi_U, \theta_T$, and ψ_T denote pitch and yaw angles from the LOS frame to the body frames of the UAV and target, respectively [29]. The LOS frame denotes a coordinate system constructed by applying rotations of λ_y and λ_z to the inertial frame, where the UAV body frame serves as the origin and the LOS direction is taken as the x-axis. In addition, the pitch and yaw angles of the UAV and target are governed by

$$\begin{aligned} \dot{\theta}_U &= \frac{a_{z_U}}{V_U} + \left(\frac{\mathbf{f}_1}{R}\right)\tan\lambda_y \sin\psi_U - \left(\frac{\mathbf{f}_2}{R}\right)\cos\psi_U, \\ \dot{\psi}_U &= \frac{a_{y_U}}{V_U\cos\theta_U} - \left(\frac{\mathbf{f}_1}{R\cos\theta_U}\right)\sin\theta_U\cos\psi_U\tan\lambda_y \\ &\quad - \left(\frac{\mathbf{f}_2}{R\cos\theta_U}\right)\sin\theta_U\sin\psi_U - \left(\frac{\mathbf{f}_1}{R}\right), \\ \dot{\theta}_T &= \frac{a_{z_T}}{V_T} + \left(\frac{\mathbf{f}_1}{R}\right)\tan\lambda_y \sin\psi_T - \left(\frac{\mathbf{f}_2}{R}\right)\cos\psi_T, \\ \dot{\psi}_T &= \frac{a_{y_T}}{V_T\cos\theta_T} - \left(\frac{\mathbf{f}_1}{R\cos\theta_T}\right)\sin\theta_T\cos\psi_T\tan\lambda_y \\ &\quad - \left(\frac{\mathbf{f}_2}{R\cos\theta_U}\right)\sin\theta_T\sin\psi_T - \left(\frac{\mathbf{f}_1}{R}\right), \\ \mathbf{f}_1 &= V_T\cos\theta_T\sin\psi_T - V_U\cos\theta_U\sin\psi_U, \\ \mathbf{f}_2 &= V_T\sin\theta_T - V_U\sin\theta_U. \end{aligned} \tag{4}$$

3. DRL-Based Guidance Law

In this section, the guidance method based on deep reinforcement learning (DRL) is proposed. To this end, the soft actor–critic (SAC) method is employed, and the SAC-based guidance law is developed.

3.1. Soft Actor–Critic

This subsection briefly describes the SAC used in this study. The SAC algorithm was developed to solve the limitations of existing policy-based reinforcement learning techniques, which have limitations such as low sample efficiency and sensitivity to hyperparameters. The SAC method uses an off-policy technique that improves sample efficiency by reusing past experiences. In addition, by using the maximum entropy model, it can be applied to the continuous action domain, and the convergence to the optimal policy is improved. The actor–critic architecture of the SAC method consists of an actor network that generates actions based on the current policy and a critic network that evaluates the actions and delivers feedback to the actor network for policy updates. The critic network consists of two Q-network and Q-target network pairs to increase convergence to the optimal policy. During the learning process, the actor and critic networks are updated through the Q objective function. The critic network estimates the Q value of the observed value–action pair based on the current policy and updates it using the temporal difference (TD) error between the estimated Q value and the observed reward. Then, the actor network is updated to maximize the expected value of the policy's reward sum based on the Q value feedback from the critic network. The objective functions for the actor and critic networks are defined as follows:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim\mathbf{D}}\left[\frac{1}{2}\left(Q_\theta(s_t,a_t) - r(s_t,a_t) + \gamma\mathbb{E}_{s_{t+1}\sim p}[V_{\bar{\theta}}(s_{t+1})]\right)^2\right],$$
$$J_\pi(\phi) = \mathbb{E}_{s_t\sim\mathbf{D}}\Big[\mathbb{E}_{a_t\sim\pi_\phi}\left[\alpha\log\left(\pi_\phi(a_t|s_t)\right) - Q_\theta(s_t,a_t)\right]\Big]. \quad (5)$$

where $\mathbb{E}_{(s_t,a_t)\sim\mathbf{D}}$ denotes the expectation over state–action pairs sampled from the replay buffer D, which contains the transition data collected during training.

If the update proceeds in the direction of improving each objective function, it can be said that the policy network approaches the optimal policy and the pseudocode of the SAC method is as in Algorithm 1.

Algorithm 1: Soft actor–critic

Input: θ_1, θ_2, ϕ: parameters;
$\bar{\theta}_1, \bar{\theta}_2$: target network weights;
$\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$: target network initialization;
$\mathbf{D} \leftarrow \emptyset$: replay buffer initialization;
for each iteration **do**
  for each environment step **do**
    $a_t \sim \pi_\phi(a_t|s_t)$;
    $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$;
    $\mathbf{D} \leftarrow \mathbf{D} \cup (s_t, a_t, r(s_t, a_t), s_{t+1})$;
  end
  for each gradient step **do**
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_\theta J_Q(\theta)\ for\ i \in 1, 2$;
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$;
    $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$;
    $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}\ for\ i \in 1, 2$
  end
end
Output: θ_1, θ_2, ϕ

3.2. SAC-Based 2D Guidance

The UAV tracking problem in the 2D environment is represented as the MDP, which is described as follows:

$$\begin{aligned}
\mathbf{S}_{2D} &= [\mathbf{S}_{U,2D}, \mathbf{S}_{T,2D}](\mathbf{S}_{i,2D} = [x_i, y_i, \psi_i], i = U, T), \\
\mathbf{O}_{2D} &= \left[\lambda_y, \dot{\lambda}_y, \frac{R_2}{R_{init,2}}, \dot{R}_2\right], \\
\mathbf{A}_{2D} &= \bar{a}_{y_U}, \\
\mathbf{R}_{2D} &= \begin{cases} 100, & \text{if } R_2 < 3 \\ -100, & \text{else if } R_2 > R_{prev}, \\ -(\frac{\text{ZEM}_2}{1e4})^2 & \text{else} \end{cases}
\end{aligned} \tag{6}$$

where the subscript 2, i.e., $(\cdot)_2$, denotes the 2D environment. The subscripts U and T denote UAV and target, respectively. The variable R_{prev} is the distance before step 1. The ZEM_2 is the zero-effort miss in the 2D environment. It is known that track performance is maximized if the ZEM_2 is approximated to zero [1]. Therefore, the reward function is set to minimize the ZEM_2 when it is not a termination condition. The ZEM_2 is formulated as follows [30,31]:

$$\text{ZEM}_2 = \frac{R_2 V_{l,2}}{\sqrt{V_{r,2}^2 + V_{l,2}^2}}, \tag{7}$$

where $V_{l,2} = R_2\dot{\lambda}_y$, $V_{r,2} = V_T\cos(\theta_T - \lambda_y) - V_U\cos(\theta_U - \lambda_y)$.

As addressed above, it is assumed that the LOS angle λ_y, the LOS angular rate $\dot{\lambda}_y$, the distance R_2, and change rate of the distance $\dot{R}_2$ are measurable. The information constitutes the observation O_2 and serves as input for the critic and actor networks, as shown in Figure 3.

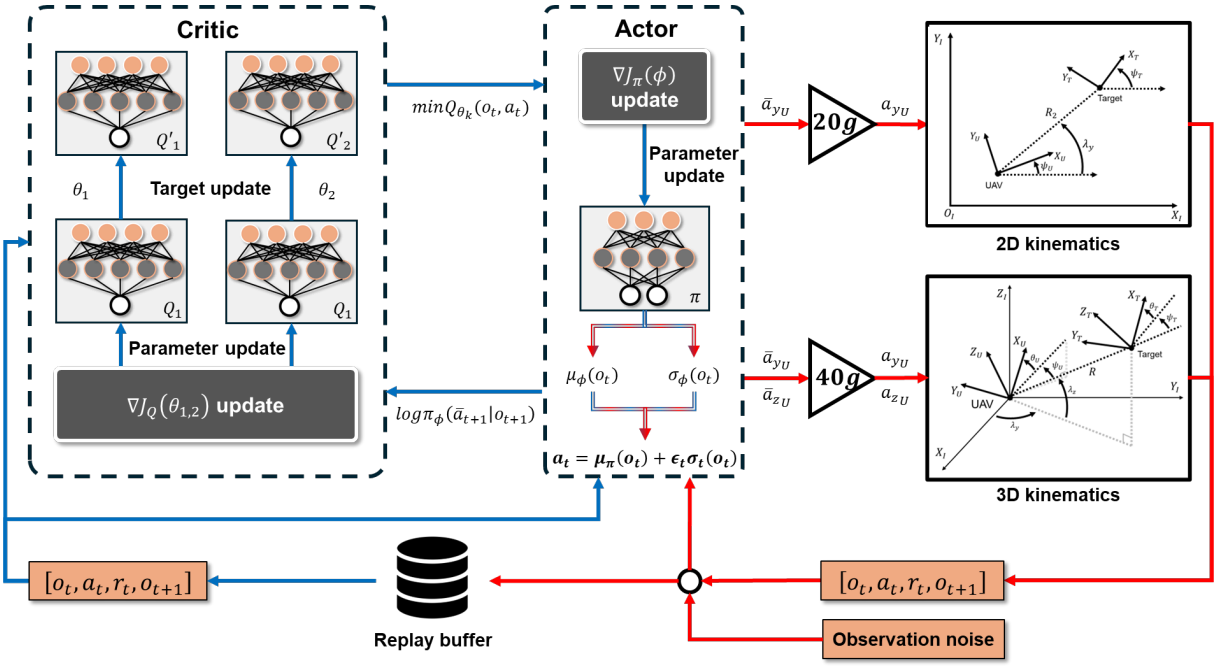


Figure 3. Overall architecture: the blue lines are flow of training, the red lines are flow of inference, and the combined blue and red lines represent both the training and inference flows.

The output of the actor network, $\bar{a}_{y_U}$, becomes a normalized guidance command for the UAV. That is, the output $\bar{a}_{y_U}$ ranges from -1 to $+1$. The final SAC-based 2D guidance

law a_{y_U} is defined as the output $\bar{a}_{y_U}$ (of the actor network) multiplied by a scale factor of 20 g. The scale factor of 20 g is adjustable depending on a control system of the UAV.

3.3. SAC-Based 3D Guidance

Unlike the 2D environmental problems, the observation values in the 3D environments include the pitch and yaw angles of the UAV. The MDP in 3D environments is defined as follows:

$$\begin{aligned}
\mathbf{S} &= [\mathbf{S}_U, \mathbf{S}_T](\mathbf{S}_i = [x_i, y_i, z_i, \theta_i, \psi_i], i = U, T),\\
\mathbf{O} &= \left[\theta_U, \psi_U, \lambda_y, \lambda_z, \dot{\lambda}_y, \dot{\lambda}_z, \frac{R}{R_{init}}, \dot{R}\right],\\
\mathbf{A} &= \left[\bar{a}_{y_U}, \bar{a}_{z_U}\right],\\
\mathbf{R} &= \begin{cases} 100, & \text{if } R < 3 \\ -100, & \text{else if } R > R_{prev} \\ -(\frac{\text{ZEM}}{1e5})^2 & \text{else} \end{cases}.
\end{aligned} \tag{8}$$

The variable R_{prev} is the distance before step 1. Similar to Equation (6), the reward function is set to minimize the ZEM, which is composed of the lateral ZEM_y and longitudinal ZEM_z. The ZEM in the 3D environment is formulated as follows:

$$\text{ZEM} = \sqrt{\text{ZEM}_y{}^2 + \text{ZEM}_z{}^2}, \tag{9}$$

where

$$\begin{aligned}
\text{ZEM}_y &= \frac{RV_{l_y}}{\sqrt{V_{r_y}^2 + V_{l_y}^2}},\\
V_{r_y} &= V_T \cos(\psi_T - \lambda_y) - V_U \cos(\psi_U - \lambda_y),\\
V_{l_y} &= V_T \sin(\psi_T - \lambda_y) - V_U \sin(\psi_U - \lambda_y),\\
\text{ZEM}_z &= \frac{RV_{l_z}}{\sqrt{V_{r_z}^2 + V_{l_z}^2}},\\
V_{r_z} &= V_T \cos(\theta_T - \lambda_z) - V_U \cos(\theta_U - \lambda_z),\\
V_{l_z} &= V_T \sin(\theta_T - \lambda_z) - V_U \sin(\theta_U - \lambda_z).
\end{aligned}$$

As addressed in the MDP in the 2D environment, it is assumed that the LOS angles λ_y, λ_z, the LOS angular rates $\dot{\lambda}_y, \dot{\lambda}_z$, the distance R, and change rate of the distance $\dot{R}$ are available. The observation vector $\mathbf{O}$ is employed as the input vector for the critic and actor networks. The action outputs $\bar{a}_{y_U}, \bar{a}_{z_U}$ by the neural networks range from -1 to $+1$ and are then multiplied by a scale factor of 40 g. The values become the acceleration commands a_{y_U} and a_{z_U}, and are delivered to the environment.

4. Validations

In this section, extensive simulations are performed to validate the performance of the proposed guidance law. Then, the results are analyzed.

4.1. Simulation Setup

This subsection describes the environment for training and validation. The actor and critic network structure of SAC in this study is shown in Table 1. All neural networks consist of an input layer, two hidden layers of 256 in size, and an output layer. The actor network

receives only the observation values as input and generates the actions, and the critic network generates Q values from the observation values and actions. The hyperparameters for the network training are shown in Table 2. The total time step required for learning in each environment is set to 10^6 and 5×10^6, respectively.

Table 1. SAC network architecture.

	Actor Network		Critic Network	
Layer	**Unit**	**Activation Function**	**Unit**	**Activation Function**
Input	4 (2D env.) 8 (3D env.)	Linear	4 (2D env.) + 1 8 (3D env.) + 2	Linear
Hidden	256	ReLU	256	ReLU
Hidden	256	ReLU	256	ReLU
Output	1 (2D env.) 2 (3D env.)	Linear	1	Linear

Table 2. Hyperparameters of SAC.

Parameter	Value
Learning rate	0.0003
Discounting factor	0.99
Buffer size	10^6
Learning start	100
Batch experience	256
Episode time steps	5000
Total time steps	2D: 10^6, 3D: 5×10^6

The 2D and 3D kinematic model-based environments are constructed using the OpenAI Gym python module. The Gym python module [28] is a popular platform for training and testing in the RL area. The 4th-order runge-kutta is used for integration with a time interval of 0.01 s in the 2D and 3D environments used for learning and verification. This integration time step setting (0.01) is used in most inductive studies [32–34]. It maintains computational efficiency and a realistic inductive update rate. It also provides sufficient numerical accuracy [32]. Especially, the observed values measured in the Gym environment are normalized to range from -1 to $+1$, and are utilized in the training and validation of the proposed method. The action output from the policy is normalized to range from -1 to $+1$ in order for the scalability increment of the proposed method. In addition, to evaluate the robustness of the proposed method against sensor uncertainty, synthetic observation noise is intentionally injected into the measurement vector before it is passed to the actor and critic networks. The default Gym-based simulation does not include sensor disturbance models, and real sensor noise characteristics (e.g., seeker LOS jitter, IMU drift, radar angle/velocity errors) are not publicly available due to military confidentiality. Therefore, we adopt a widely used and physically reasonable assumption in the guidance literature by modeling measurement disturbances as zero-mean Gaussian noise.

$$\begin{aligned} \tilde{o} &= o + \mathbf{N}(0, \sigma), \\ \sigma &\in [0.001, 0.01, 0.05], \end{aligned} \tag{10}$$

where o is the noise-free observation generated from the kinematic environment. This noise is added immediately before feeding the state vector into both the actor and critic networks, ensuring that the learned policy remains robust to corrupted measurements.

The noise levels used in this study follow two approaches: Training—a mild noise level of $\sigma = 0.001$ is applied to help the policy learn stable feature extraction without destabilizing early training. Evaluation—all three noise levels (0.001, 0.01, 0.05) are tested to assess robustness under progressively harder sensing conditions, with $\sigma = 0.05$ representing a disturbance 50 times stronger than the training condition.

First, the DRL-based guidance method is validated in a 2D environment. This confirms that the overall framework is properly structured. Next, the proposed method is extended to a 3D environment, and its performance is verified. Each procedure is performed with the settings shown in Tables 3 and 4.

Table 3. Initial conditions for each phase: (□) randomly selected at 10 deg intervals, (△) randomly selected at 1 deg intervals.

	Initial Condition (2D Environment)					
Phase	$[X_{U,0}, Y_{U,0}]$ **(m, m)**	$[X_{T,0}, Y_{T,0}]$ **(m, m)**	V_U **(m/s)**	V_T **(m/s)**	ψ_U **(deg)**	ψ_T **(deg)**
Training	[0, 0]	[10,000, 0]	500	200	−40–40 (□)	−40–40 (□)
Evaluation	[0, 0]	[10,000, 0]	500	200	−90–90 (△)	−90–90 (△)

Table 4. Initial conditions for each phase: (□) randomly selected at 10 deg intervals, (△) randomly selected at 1 deg intervals; $[X_{T,r}, Y_{T,r}, Z_{T,r}]$ is randomly selected from the range 3000–7000, respectively.

	Initial Condition (3D Environment)					
Phase	$[X_{U,0}, Y_{U,0}, Z_{U,0}]$ **(m, m, m)**	$[X_{T,0}, Y_{T,0}, Z_{T,0}]$ **(m, m, m)**	V_U **(m/s)**	V_T **(m/s)**	θ_U, ψ_U **(deg)**	θ_T, ψ_T **(deg)**
Training	[0, 0, 0]	[5000, 5000, 5000]	900	700	45	−40–40 (□)
Evaluation 1	[0, 0, 0]	[5000, 5000, 5000]	900	700	45	−90–90 (△)
Evaluation 2	[0, 0, 0]	$[X_{T,r}, Y_{T,r}, Z_{T,r}]$	900	700	45	−90–90 (△)
Evaluation 3	[0, 0, 0]	$[X_{T,r}, Y_{T,r}, Z_{T,r}]$	900	700	0–90 (△)	−90–90 (△)

In the Tables, the tracking scenario is defined using the kinematic model and the generation of guidance commands. The control effort required for command tracking is not considered. Since system dynamics are not modeled, aerodynamic effects such as Mach number variation and altitude-dependent behavior are neglected.

In this study, the UAV is modeled using the kinematic tracking framework. The objective is to evaluate the guidance algorithm independently of a specific airframe or propulsion system. The maneuvering capability is represented by bounded lateral acceleration limits (±20 g in 2D and ±40 g in 3D). These limits define the achievable turn capability. This abstraction enables fair evaluation under controlled constraints.

4.2. Validation in 2D Environment

In this subsection, validation is performed in the 2D environment and the results are analyzed. The proposed method in the 2D tracking environment is validated through an evaluation phase using a DRL model that has undergone a learning phase. The initial condition for the learning and evaluation phases are shown in Table 3. Since the main focus of this study is on the 3D tracking environment, the 2D environment validation was conducted to verify the normal operation of the entire framework.

The objective of the proposed method is to reduce the final miss distance to within 3 m. If the final distance between the UAV and the target is below this threshold, the trial is considered successful. Conversely, if the distance exceeds 3 m or the tracking takes more than 40 s, it is counted as a failure. The observations used in the evaluation phase contain

Gaussian noise ($\mu = 0, \sigma = 0.001, 0.01, 0.05$). In contrast, the training phase uses a smaller standard deviation of 0.001.

All methods were evaluated using 1000 Monte Carlo simulations under randomized initial conditions. The figures show only representative trajectories for clarity. The reported success rates and statistical results are computed from the full set of simulations.

As shown in the Figures 4–6, the evaluation results demonstrate the robustness and generalization capability of the proposed DRL-based guidance law. In the trajectory and state plots, each colored solid line corresponds to an individual simulation episode, and curves with the same color represent the trajectory and associated state evolution of that episode. This representation allows consistent interpretation of the guidance behavior under varying initial conditions and noise realizations. Although the policy was trained under a limited range of initial yaw angles (±40°, with 10° intervals), the evaluation phase tested a much wider range (±90°, with 1° intervals). Despite this extrapolation to previously unseen states, the method consistently achieved success rates above 90%. This indicates that the learned policy effectively captures general guidance principles rather than simply overfitting to specific training conditions.

In addition, robustness to observation noise was confirmed. As the standard deviation of the Gaussian noise increased from 0.001 to 0.05, the performance degradation was minimal, with only a marginal drop in success rate. This demonstrates that the SAC-based policy remains stable under noisy sensor conditions and suggests its potential for real-world deployment.

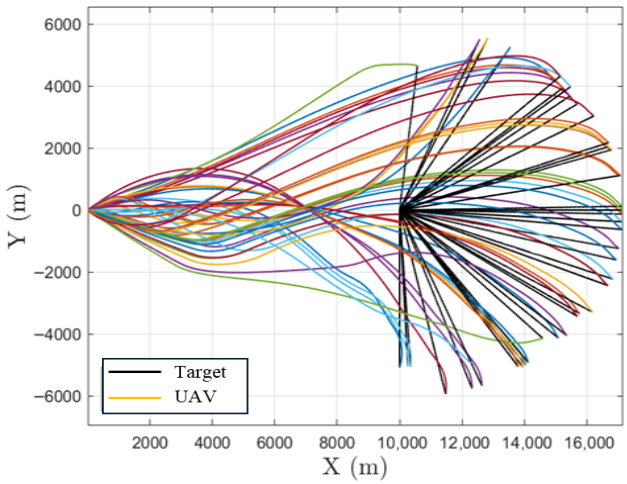


Figure 4. Trajectories of the UAV and target in 2D environment with noise (**N**(1, 0.001)).

The acceleration command output from the actor network was also analyzed. As shown in Figure 3, the network's output is normalized between -1 and $+1$, then scaled by a factor of 20 g to generate the final control command. As shown in Figure 4, the acceleration command is relatively smooth and continuous. This suggests that the DRL-based model can sufficiently demonstrate physically realistic behavior and numerically stable control performance. A closer analysis reveals that while there is some chattering in the acceleration command, this command is ultimately a guidance command in a system that is divided into guidance and control. Therefore, it is sufficiently surmountable during the control.

Meanwhile, Figures 4–6 show the verification results with a noise variance of 0.001, i.e., the noise condition applied during training. Similar behavior was observed even when noise levels of 0.001 and 0.05 were applied, with action chattering increasing by approximately 2 to 2.5 times. The decrease in the success rate as noise increased was observed to be around 2%.

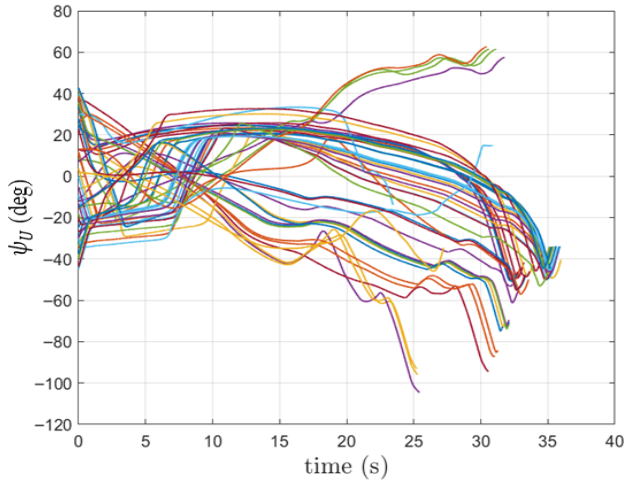


Figure 5. Yaw angle of the UAV in 2D environment with noise (**N**(0, 0.001)).

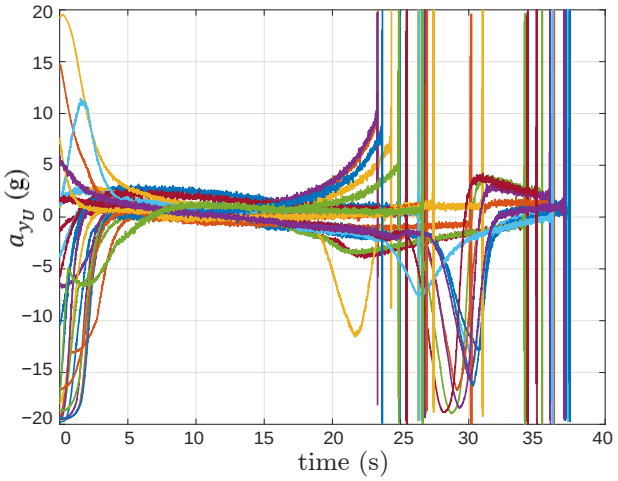


Figure 6. Control input a_{y_U} history of the UAV in 2D environment with noise (**N**(0, 0.001)).

Furthermore, as shown in Figure 6, the reason a_{y_U} increases rapidly when reaching the end point is related to the LOS rate ($\dot{\lambda}_2$). That is, according to Equation (2), as the distance between the UAV and the target decreases, the $\dot{\lambda}_2$ increases. This causes a_{y_U} for tracking at the end point to change significantly.

In summary, the proposed SAC-based DRL guidance framework demonstrates high generalization ability and robustness. Despite training under limited conditions, it maintains consistent performance even under a wide range of initial conditions and noise. Furthermore, the output acceleration commands are confirmed physically feasible. Through this, the overall framework of the proposed method has been validated in the 2D environment. Therefore, the framework is used to extend and validate the proposed method in the 3D environment.

4.3. Validation in 3D Environment

In this subsection, validation is performed in the 3D environment and the results are analyzed. Similar to the validation in the 2D environment, the proposed guidance law is trained and validated in the 3D environment. Because the 3D environment is more challenging than the 2D environment, the evaluation for the 3D environment is performed in three phases, as shown in Table 4.

The success criterion is consistent with the 2D case: if the final distance between UAV and target is less than 3 m, it is considered a success. If the UAV diverges or the tracking

takes more than 40 s, it is considered a failure. Gaussian noise $(\mu = 0, \sigma = 0.001, 0.01, 0.05)$ is added to the observation values just before they are passed into the policy network.

The evaluation results in the 3D environment confirm the generalization capability and robustness of the proposed SAC-based guidance law. The evaluation progresses from phase 1 to phase 3. As a result, the initial conditions become increasingly diverse. This is especially true in phase 3, where the UAV's direction is randomized. Despite this, as shown in Table 5, the proposed method maintains consistently high success rates above 90% in phases 1 and 2, and over 80% even in phase 3. This demonstrates that the learned policy is not overfitted to a narrow set of initial conditions, but can effectively adapt to unseen configurations in a high-dimensional state space.

In terms of noise robustness, only a minimal degradation in success rate (about 1–2%) was observed when the standard deviation of the Gaussian noise increased from 0.01 to 0.05. This indicates that the policy network learned to extract stable and effective control behavior even in the presence of substantial observation noise, reinforcing the practical applicability of the method under sensor uncertainty.

The control actions generated by the policy network were also analyzed. The outputs of the actor network are normalized in the range $[-1, +1]$ and scaled by a factor of 40 g to produce actual acceleration commands. As shown in Figures 7–9c, both a_{y_U} and a_{z_U} commands remain smooth and continuous over time, without abrupt oscillations or instability. This suggests that the learned policy respects the underlying UAV kinematics and produces physically meaningful control commands suitable for real-time use.

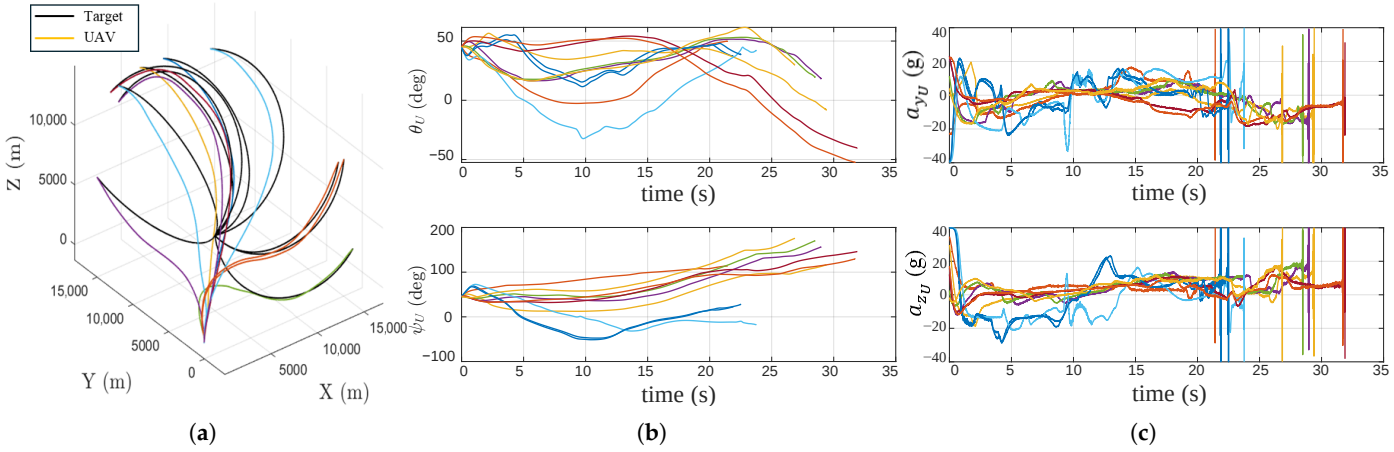


Figure 7. Validation results (evaluation phase 1) in 3D environment with noise (**N**(0, 0.001)): (**a**) trajectories of the UAV and target, (**b**) pitch and yaw angles of the UAV, and (**c**) deep neural network-based guidance laws.

To benchmark the proposed approach, simulations were conducted using the traditional PN law under identical scenarios and noise conditions. All guidance methods used the same UAV kinematic model, velocity, and acceleration limits. This ensures a fair comparison. Therefore, performance differences are attributed to the guidance algorithms rather than airframe capability.

The terminal distance threshold is an important evaluation criterion. As shown in Table 5, both methods achieve similar success rates under relaxed terminal conditions (e.g., 5 m). In this case, precise terminal correction is not required. Conventional guidance laws can perform adequately. However, differences appear under stricter terminal conditions (e.g., 3 m and 2 m). PN generates acceleration commands proportional to the LOS angular rate. As the UAV approaches the target, the LOS rate increases. This amplifies the control input. As a result, terminal accuracy degrades and success rates decrease.

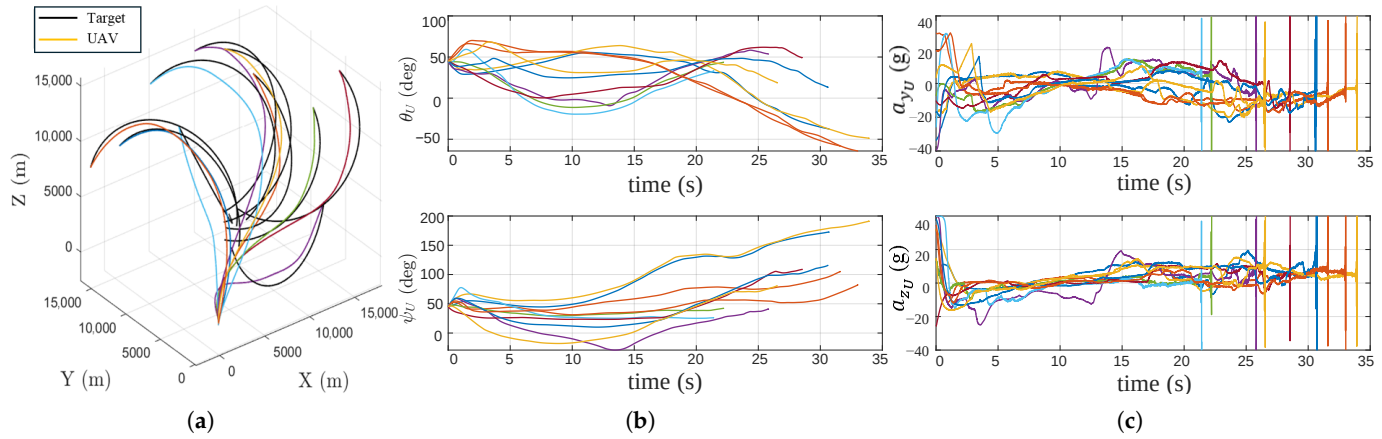


Figure 8. Validation results (evaluation phase 2) in 3D environment with noise (**N**(0, 0.001)): (**a**) trajectories of the UAV and target, (**b**) pitch and yaw angles of the UAV, and (**c**) deep neural network-based guidance laws.

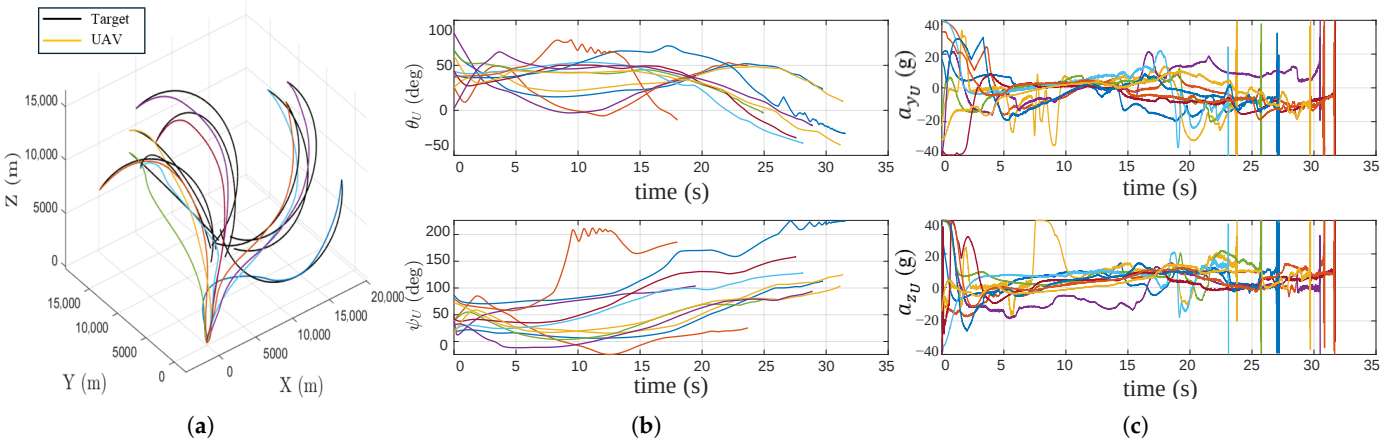


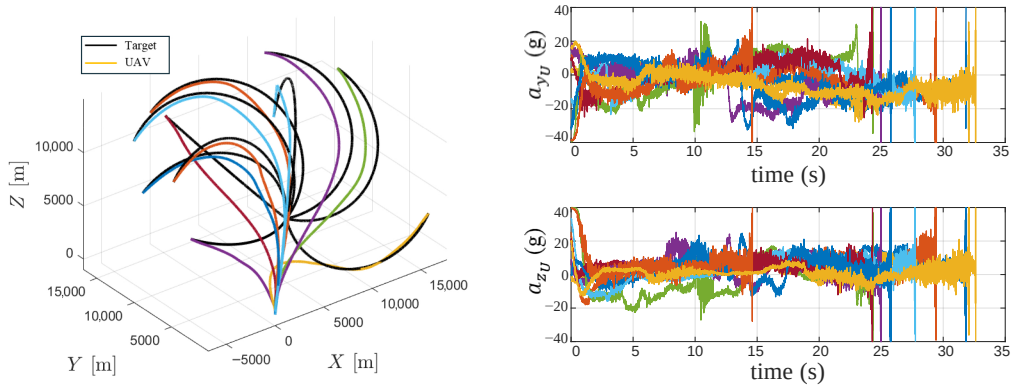
Figure 9. Validation results (evaluation phase 3) in 3D environment with noise (**N**(0, 0.001)): (**a**) trajectoriesof the UAV and target, (**b**) pitch and yaw angles of the UAV, and (**c**) deep neural network-based guidance laws.

In contrast, the proposed method is trained to minimize the ZEM. The ZEM represents the predicted terminal miss distance. This allows the policy to maintain effective corrective control near tracking. Therefore, the proposed method achieves higher success rates under strict terminal conditions and observation noise. In summary, a comprehensive performance evaluation was conducted using multiple terminal distance thresholds. This demonstrates that the proposed method remains effective under strict terminal precision requirements.

In addition to the success rates summarized in Table 5, the performance analysis also includes statistical measures computed over 1000 Monte Carlo runs for each noise level. For the SAC-guided trackings, the terminal miss distance achieved an average of 1.52 ± 0.41 m, while the tracking time converged to 24.7 ± 3.9 s across all successful trials. These results confirm that the proposed policy not only maintains high success rates exceeding 90% under all tested noise conditions, but also achieves low variability in both terminal accuracy and tracking duration, even when the observation noise is increased by a factor of 50.

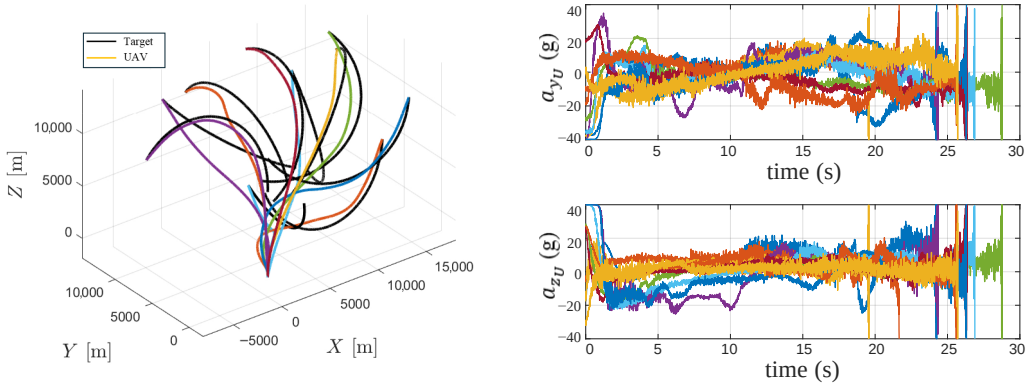
To analyze the performance of the proposed method more closely, trajectories and actions are plotted when the noise variance is 0.05. As shown in Figures 10–12, it is confirmed that the target is effectively shot down even though the variance of noise is increased by 50 times compared to the variance applied during training (0.001). In contrast

to the significantly increased noise variance, the chattering of the actions derived from the DRL model is relatively small. The proposed technique effectively counters noise, and further training using reward shaping to attenuate action chattering is expected to yield significantly better performance.



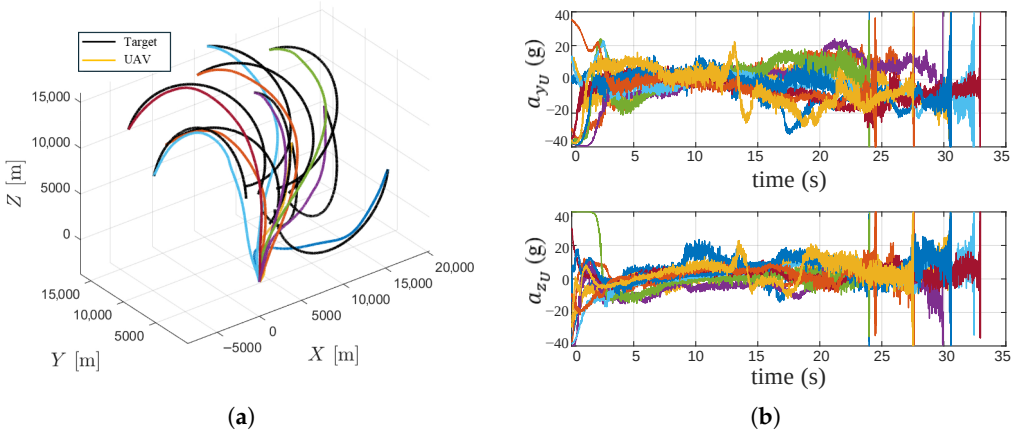
(**a**) (**b**)

Figure 10. Validation results (evaluation phase 1) in 3D environment with noise (**N**(0, 0.05)): (**a**) trajectories of the UAV and target, (**b**) pitch and yaw angles of the UAV.



(**a**) (**b**)

Figure 11. Validation results (evaluation phase 2) in 3D environment with noise (**N**(0, 0.05)): (**a**) trajectories of the UAV and target, (**b**) pitch and yaw angles of the UAV.



(**a**) (**b**)

Figure 12. Validation results (evaluation phase 3) in 3D environment with noise (**N**(0, 0.05)): (**a**) trajectories of the UAV and target, (**b**) pitch and yaw angles of the UAV.

Table 5. Success rate of the proposed method and PN guidance.

		Proposed		PN Guidance	
Evaluation Phase	**End Distance (m)**	**N (1, 0.01)**	**N (1, 0.05)**	**N (1, 0.01)**	**N (1, 0.05)**
1	5	93.2%	92.5%	97.9%	94.2%
	4	92.9%	92.1%	95.7%	89.6%
	3	93.0 %	91.9%	89.8%	79.6%
	2	91.1%	90.7%	74.4%	62.3%
2	5	92.3%	91.3%	95.1%	90.7%
	4	92.1%	90.8%	91.8%	86.2%
	3	91.8%	90.6%	84.9%	76.4%
	2	89.5%	89.8%	70.5%	59.2%
3	5	83.6%	82.9%	88.3%	84.5%
	4	83.6%	82.6%	86.0%	80.0%
	3	83.3%	82.1%	79.2%	70.7%
	2	80.9%	79.5%	65.4%	56.2%

In summary, it was confirmed that the proposed DRL-based guidance method guarantees similar performance to the PN-based guidance law in the environments where observation values are mixed with noise, and can be effectively used for missions that require a closer distance to the target.

4.4. Real-Time Implementability

The feasibility of deploying the learned policy on embedded avionics is evaluated by examining the network size, per-cycle computational load, and compatibility with real-time processors typically used in aerospace and defense-grade embedded systems. The actor network contains an input layer (4 units in 2D, 8 units in 3D), two fully connected hidden layers with 256 neurons each, and an output layer of 1–2 units.

For the 3D case, the total parameter count is input to Hidden 1: $8 \times 256 + 256 = 2304$, Hidden 1 to Hidden 2: $256 \times 256 + 256 = 65,792$, Hidden 2 to Output: $256 \times 2 + 2 = 514$, total parameter $= 68,618$. The 2D actor contains approximately 66,000 parameters. This corresponds to a memory footprint of about 270 kB in 32-bit floating point format, which is well within the limits of compact embedded avionics architectures.

The computational cost per forward pass is also modest. For the 3D actor network, MAC (multiply-accumulate) operations $= (8 \times 256) + (256 \times 256) + (256 \times 2) = 70,000$, equivalent to roughly 140 kFLOPs when counting multiply and addition separately. At a conservative 100 Hz guidance update rate, the total computational load becomes $140,000 \times 100 = 14$ MFLOPs/s, which is significantly below the processing capability of widely available real-time embedded processors used in applications.

- TI C6000 floating-point DSPs provide approximately 1 GFLOPS of single-precision performance (e.g., TMS320C6701 at 167 MHz) [35].
- ARM Cortex-R5/R7–based TI Hercules MCUs deliver hundreds of MFLOPS, with peak figures around 300–330 MFLOPS depending on configuration [36].

These devices exceed the required throughput by more than an order of magnitude, demonstrating that the actor network can be executed comfortably within real-time constraints on commercially available embedded hardware. Importantly, only the inference stage of the SAC policy runs on the embedded processor. All computationally intensive operations—critic updates, backpropagation, and replay-buffer sampling are performed offline on a workstation. This offline–online separation ensures that the high training cost does not burden the onboard avionics.

Based on these considerations, the proposed DRL-based guidance policy is expected to meet real-time implementation requirements and be readily exploitable on modern embedded processors used in aerospace and defense systems.

5. Conclusions

In this study, a reinforcement learning-based guidance law was proposed to overcome the limitations of conventional methods under uncertain and noisy environments. The proposed method formulates both two-dimensional and three-dimensional tracking scenarios as Markov decision processes (MDPs). And then, the policy was trained using the soft actor–critic (SAC) algorithm to directly generated acceleration commands from noisy observations.

Unlike traditional guidance laws such as proportional navigation (PN), which relied solely on line-of-sight (LOS) angular rate, the proposed method utilized a richer set of state variables. These included relative distance, LOS angle, and angular rate, which allowed it to effectively minimize the zero-effort miss (ZEM), especially in close-range trackings. The training environment incorporates Gaussian noise to simulate realistic sensor uncertainty, enabling the learned policy to generalize beyond its training distribution.

Extensive validation using the OpenAI gym-based tracking simulator showed that the proposed method consistently outperformed the PN baseline. This performance advantage was especially evident under strict terminal constraints and high levels of observation noise. In both 2D and 3D simulations, the learned policy achieved high success rates, which confirmed its robustness, adaptability, and potential for practical deployment in the UAV guidance systems.

Limitation and Future Work

The proposed SAC-based guidance framework shows strong performance in kinematic-level simulations. However, several limitations remain.

Observation disturbances are modeled as Gaussian noise. This does not fully represent real-world disturbances such as wind gusts, aerodynamic uncertainties, and actuator dynamics. In addition, aerodynamic nonlinearities and actuator saturation are not considered. Dynamic coupling between guidance and control is also neglected. These factors may affect performance under realistic flight conditions.

Future work will extend the framework to higher-fidelity environments. This includes aerodynamic effects, actuator dynamics, and environmental disturbances. Hardware-in-the-loop simulation tests will also be conducted to evaluate practical applicability.

Author Contributions: Methodology, S.K.; Investigation, J.S.; Writing—original draft, S.K.; Writing—review and editing, J.S. and H.-G.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by ** Artificial Intelligence Research Laboratory for Flight Control ** funded by the Agency for Defense Development and Defense Acquisition Program Administration under Grant UD230014SD and by the Regional Innovation System & Education (RISE) program through the (Chungbuk Regional Innovation System & Education Center), funded by the Ministry of Education (MOE) and the (Chungcheongbuk-do), Republic of Korea (2026-RISE-11-014-01).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Reardon, B.E.; Lloyd, J.M.; Perel, R.Y. Tuning missile guidance and control algorithms using simultaneous perturbation stochastic approximation. *Johns Hopkins APL Tech. Dig.* **2010**, *29*, 85.
2. Fu, Z.; Dai, Y.; Zhang, K. Research progress on design methods for missile integrated guidance and control. In *Proceedings of the 2017 International Conference on Automation, Control and Robots*; Association for Computing Machinery: New York, NY, USA, 2017; pp. 15–19.
3. Liang, X.; Xu, B.; Jia, K.; Liu, X. Adaptive NN control of integrated guidance and control systems based on disturbance observer. *J. Frankl. Inst.* **2023**, *360*, 65–86. [CrossRef]
4. Sreeja, S.; Hablani, H. Precision munition guidance and moving-target estimation. *J. Guid. Control Dyn.* **2016**, *39*, 2100–2111. [CrossRef]
5. Satir, A.S.; Demir, U.; Sever, G.G.; Ure, N.K. Nonlinear model based guidance with deep learning based target trajectory prediction against aerial agile attack patterns. In *Proceedings of the 2021 American Control Conference (ACC)*; IEEE: New York, NY, USA, 2021; pp. 2607–2612.
6. Ma, S.; Wang, X.; Wang, Z. Field-of-view constrained impact time control guidance via time-varying sliding mode control. *Aerospace* **2021**, *8*, 251. [CrossRef]
7. Lu, F.; Chen, Z.; Wang, K. Nonlinear Optimal Guidance for Impact Time Control with Field-of-View Constraint. *arXiv* **2025**, arXiv:2503.15362.
8. Ming, C.; Wang, X. Nonsingular terminal sliding mode control-based prescribed performance guidance law with impact angle constraints. *Int. J. Control Autom. Syst.* **2022**, *20*, 715–726. [CrossRef]
9. Luo, H.; Liu, Z.; Jin, T.; Lee, C.H.; He, S. Computational Predictor–Corrector Homing Guidance for Constrained Impact. *J. Guid. Control Dyn.* **2025**, *48*, 1366–1380. [CrossRef]
10. Kwon, H.H.; Choi, H.L. A convex programming approach to mid-course trajectory optimization for air-to-ground missiles. *Int. J. Aeronaut. Space Sci.* **2020**, *21*, 479–492. [CrossRef]
11. Chai, R.; Tsourdos, A.; Savvaris, A.; Chai, S.; Xia, Y.; Chen, C.P. Review of advanced guidance and control algorithms for space/aerospace vehicles. *Prog. Aerosp. Sci.* **2021**, *122*, 100696. [CrossRef]
12. Kim, M.; Hong, D.; Park, S. Deep neural network-based guidance law using supervised learning. *Appl. Sci.* **2020**, *10*, 7865. [CrossRef]
13. Hongxia, L.; Huijie, L.; Yuanli, C. Efficient and accurate online estimation algorithm for zero-effort-miss and time-to-go based on data driven method. *Chin. J. Aeronaut.* **2019**, *32*, 2311–2323.
14. He, S.; Shin, H.S.; Tsourdos, A. Computational missile guidance: A deep reinforcement learning approach. *J. Aerosp. Inf. Syst.* **2021**, *18*, 571–582. [CrossRef]
15. Kirk, R.; Zhang, A.; Grefenstette, E.; Rocktäschel, T. A survey of generalisation in deep reinforcement learning. *J. Artif. Intell. Res.* **2023**, *76*, 201–264. [CrossRef]
16. Xie, R.; Jin, X.; Zhao, Q.; Zhang, Y.; Wang, Z. Rapid Bootstrapping of Deep Reinforcement Learning with Curriculum and Imitation Strategies for Missile Guidance. *Int. J. Aeronaut. Space Sci.* **2025**, *26*, 2677–2694. [CrossRef]
17. Sumiea, E.H.; Abdulkadir, S.J.; Alhussian, H.S.; Al-Selwi, S.M.; Alqushaibi, A.; Ragab, M.G.; Fati, S.M. Deep deterministic policy gradient algorithm: A systematic review. *Heliyon* **2024**, *10*, e30697. [CrossRef]
18. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347. [CrossRef]
19. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the International Conference on Machine Learning*; PMLR: Cambridge, MA, USA, 2018; pp. 1861–1870.
20. Candeli, A.; De Tommasi, G.; Lui, D.G.; Mele, A.; Santini, S.; Tartaglione, G. A deep deterministic policy gradient learning approach to missile autopilot design. *IEEE Access* **2022**, *10*, 19685–19696. [CrossRef]
21. Hong, D.; Kim, M.; Park, S. Study on reinforcement learning-based missile guidance law. *Appl. Sci.* **2020**, *10*, 6567. [CrossRef]
22. Ahn, J.; Shin, J.; Kim, H.G. Deep reinforcement learning-based robust missile guidance. In *Proceedings of the 2022 22nd International Conference on Control, Automation and Systems (ICCAS)*; IEEE: New York, NY, USA, 2022; pp. 927–930.
23. Qiu, X.; Lai, P.; Gao, C.; Jing, W. Recorded recurrent deep reinforcement learning guidance laws for intercepting endoatmospheric maneuvering missiles. *Def. Technol.* **2024**, *31*, 457–470. [CrossRef]
24. Peng, C.; Zhang, H.; He, Y.; Ma, J. State-following-kernel-based online reinforcement learning guidance law against maneuvering target. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, *58*, 5784–5797. [CrossRef]
25. Wang, W.; Wu, M.; Chen, Z.; Liu, X. Integrated guidance-and-control design for three-dimensional interception based on deep-reinforcement learning. *Aerospace* **2023**, *10*, 167. [CrossRef]
26. Gaudet, B.; Furfaro, R. Missile homing-phase guidance law design using reinforcement learning. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Minneapolis, MN, USA, 13–16 August 2012; p. 4470.

27. Garcia, F.; Rachelson, E. Markov decision processes. In *Markov Decision Processes in Artificial Intelligence*; Wiley: Hoboken, NJ, USA, 2013; pp. 1–38.
28. Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; Zaremba, W. Openai gym. *arXiv* **2016**, arXiv:1606.01540. [CrossRef]
29. Song, S.H.; Ha, I.J. A Lyapunov-like approach to performance analysis of 3-dimensional pure PNG laws. *IEEE Trans. Aerosp. Electron. Syst.* **1994**, *30*, 238–248. [CrossRef]
30. He, S.; Song, T.; Lin, D. Impact angle constrained integrated guidance and control for maneuvering target interception. *J. Guid. Control. Dyn.* **2017**, *40*, 2653–2661. [CrossRef]
31. Rawling, A. On nonzero miss distance. *J. Spacecr. Rocket.* **1969**, *6*, 81–83. [CrossRef]
32. Cai, G.; Zhang, Y.; Li, J.; Wang, X. Research and Flight Test on the Terminal Guidance of Tactical Missiles. *Aerospace* **2024**, *11*, 975. [CrossRef]
33. Wang, Y.; Liu, Z.; Chen, M. Fuzzy Neural Sliding Mode Guidance Law with Impact Angle Constraint. *Int. J. Aerosp. Eng.* **2020**, *2020*, 4597937.
34. Liu, S.; Zhao, Y.; Chen, W. Missile Guidance Law Design Based on Free-Time Convergent Error Dynamics. *arXiv* **2023**, arXiv:2308.04743. [CrossRef]
35. Texas-Instrumetns-Incorporated. Highest Performance Floating-Point Digital Signal Processor. Available online: https://www.ti.com.cn/lit/ds/symlink/tms320c6701.pdf (accessed on 26 February 2026).
36. Texas-Instrumetns-Incorporated. TI Introduces Fastest Hercules. Available online: https://www.ti.com/ (accessed on 26 February 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.