Comparative Evaluation of Deep Reinforcement Learning Algorithms for Quadcopter Control

Mohammadbagher Shahmir[1*] and Fereshte Dehghani[1]

[1*]Faculty of Electrical and Computer Engineering, University of Kashan, Kashan, Iran.

*Corresponding author(s). E-mail(s): mohammadbaghershahmir@gmail.com;
Contributing authors: fdehghani@kashanu.ac.ir;

Abstract

Nonlinear dynamics, strong coupling between translational and rotational motion, and high sensitivity to external disturbances make precise and robust quadcopter control challenging. While classical controllers (e.g., PID) perform adequately in nominal conditions, dynamic environments and modeling uncertainties often degrade their reliability. Deep reinforcement learning (DRL) provides an adaptive, model-free alternative that learns control policies directly from interaction with a simulated environment; however, DRL methods differ significantly in stability, convergence speed, and computational cost.

This paper presents a rigorous comparative evaluation of five distinct DRL algorithms for quadcopter control, including DDPG, PPO, SAC, SD3, and the proposed ETGL-DDPG framework. We implement all methods under identical environmental constraints within a high-fidelity quadcopter simulation operating in a continuous action space (motor RPMs). Our optimization relies on a carefully scaled reward function designed to encourage target tracking while penalizing positional and angular deviations. We assess the comparative paradigms quantitatively using complementary evaluation metrics, including cumulative reward evolution across multiple independent random seeds, learning stability indicators, convergence velocity, and 3D flight trajectory quality.

Experimental evaluations demonstrate that on-policy PPO improves learning stability relative to vanilla DDPG but converges more slowly, while SAC achieves superior reward levels with noticeable policy variability. Computational and empirical analyses show that SD3 accelerates convergence and enhances tracking stability compared to the baseline methods. Crucially, ETGL-DDPG consistently attains the most stable learning dynamics, the highest performance ceiling, and

the most precise 3D tracking trajectories. These findings highlight the critical importance of enhanced exploration and long-horizon return propagation mechanisms for reliable continuous quadcopter control setups.

Keywords: Quadcopter Control, Deep Reinforcement Learning, Continuous Control, DDPG, PPO, Soft Actor-Critic, SD3, ETGL-DDPG

1 Introduction

Quadcopters, alternatively designated as four-rotor drones, have emerged as one of the primary and widely utilized configurations of Unmanned Aerial Systems (UAS) in recent years. Researchers and industries have found significant applications for these platforms across various commercial and research domains, including precision agriculture, high-resolution aerial imaging, search and rescue operations, environmental monitoring, package delivery, and urban infrastructure surveillance. Their unique capacity to deliver efficient solutions—particularly in optimizing agricultural yield, facilitating emergency logistics, and enhancing tracking accuracy in complex environments—positions them as pivotal tools in modern technological advancements [1, 2]. Additionally, recent progress in artificial intelligence, machine learning, and wireless communication protocols further expands the operational envelope of these drones in executing optimized, multi-agent tasks.

However, achieving precise and robust autonomous flight control remains a major obstacle in automatic control engineering. These challenges originate from the inherently non-linear, underdetermined, and tightly coupled six-degrees-of-freedom (6-DOF) rigid-body dynamics of the aircraft, combined with an extreme sensitivity to external aerodynamic disturbances and wind turbulence [3]. Consequently, developing advanced control laws capable of stabilizing the vehicle under highly dynamic or unforeseen operational envelopes remains a critical research frontier.

Traditional control frameworks, such as proportional-integral-derivative (PID) loops, frequently struggle when deployed in volatile real-world environments because they rely on localized linear approximations and fundamentally fail to reject unmodeled external wind vectors. While classical linear controllers exhibit adequate tracking under nominal, low-velocity conditions, their performance typically degrades into severe transient oscillations or instability during aggressive maneuvers. In contrast, reinforcement learning (RL) offers an adaptive, model-free alternative. By dynamically adjusting actuator commands based on real-time kinematic feedback, RL-driven control architectures achieve superior disturbance rejection, faster transient response times, and enhanced safety margins under complex flight conditions [4]. This robust real-time adaptability of reinforcement learning is fundamentally rooted in its capacity to incrementally process continuous streams of environmental observations and adapt policies via sequential stochastic gradient descent, effectively preventing catastrophic performance degradation when confronted with sudden dynamic shifts [5].

When integrated with deep neural networks—which are increasingly recognized as foundational tools driving modern control, fault-tolerance, and operational management methodologies across the broader aerospace industry [6] DRL excels at mapping continuous, high-dimensional state spaces directly into continuous action spaces (e.g., motor speeds), bypassing the need for rigorous aerodynamic modeling. Despite its immense potential for model-free continuous control, onboarding DRL onto physical UAVs presents severe practical challenges [7]. The optimization space remains inherently multi-objective, requiring a delicate numerical balance between trajectory tracking accuracy, angular stability, and energy conservation. Furthermore, the vast continuous action domains often lead to slow training convergence, severe value overestimation bias, and catastrophic forgetting during early exploration phases, where poor exploration policies frequently trigger boundary violations and premature episode terminations [8].

To address these challenges and provide a definitive benchmarking reference, this paper presents a rigorous comparative implementation and evaluation of five prominent continuous DRL paradigms: Soft Actor-Critic (SAC) [9], Proximal Policy Optimization (PPO) [10], Deep Deterministic Policy Gradient (DDPG) [11], Softmax Deep Double Deterministic Policy Gradient (SD3) [12], and the recently developed Enhanced Temporal Gradient Longest-step DDPG (ETGL-DDPG) framework [13]. We benchmark all evaluated algorithms under strictly identical environmental initializations and hyperparameter equity within a specialized Python flight simulation envelope.

The primary contribution of this work is a multi-dimensional quantitative analysis focusing on sample efficiency, learning stability, inference latency, and 3D geometric path quality. We dedicate special emphasis to analyzing the structural mechanisms of the ETGL-DDPG paradigm. By incorporating a goal-conditioned dual replay buffer to isolate highly successful trajectories and leveraging a longest-step return formulation for accelerated credit assignment, this framework effectively mitigates sparse-reward bottlenecks. The empirical findings established in this study offer critical design principles for compiling reliable, continuous DRL flight software. To support open science and verify full empirical reproducibility, we host and publicly provide the complete Python source code implementation for all five benchmarks at https://github.com/mohammadbaghershahmir/Quadcopter-ETGL-DDPG.

2 Related Work

In this section, we review prior studies across three main directions: classical and model-based control methods for quadcopters, deep reinforcement learning approaches for UAV control, and recent efforts aimed at improving stability and exploration in continuous-control reinforcement learning. This structured review highlights existing limitations and clarifies the research gap that our study addresses.

2.1 Robust Model-Based Quadcopter Control

Classical quadrotor control has traditionally relied on model-based designs such as PID cascades, LQR-like stabilizers, and nonlinear controllers that explicitly exploit

the system dynamics. Early foundational work established practical modeling and control pipelines for micro-quadrotors, demonstrating stable hover and trajectory regulation under structured assumptions [14, 15]. More recent studies emphasize robustness against uncertainty and disturbances using nonlinear techniques such as adaptive backstepping and sliding-mode control (SMC). For example, researchers have combined adaptive backstepping with SMC to achieve accurate trajectory tracking under parametric variations and external disturbances while maintaining low computational cost [16]. Furthermore, recent advances published within soft computing literature emphasize that the complex parameter tuning required for such robust non-linear controllers typically necessitates integration with advanced metaheuristic optimization frameworks to guarantee dynamic stability under heavy external anomalies [17]. In parallel, investigators continue to explore disturbance-aware and hybrid robust controllers, reflecting the persistent gap between idealized models and the highly coupled, nonlinear, and disturbance-sensitive reality of quadrotor flight.

2.2 Deep Reinforcement Learning for Continuous UAV Control

To overcome modeling limitations and improve adaptability, researchers increasingly adopt DRL for UAV control in continuous action spaces. Actor–critic methods such as DDPG [11], PPO [10], and SAC [9] have become standard baselines because they learn nonlinear policies directly from interaction data. For quadrotor navigation and tracking, developers have applied DDPG-based solutions to path-following and reactive obstacle avoidance, showing that learned policies generalize to practical flight tasks when suitable training pipelines and sensing configurations support them [18, 19]. Beyond pure DRL, hybrid schemes that blend robust control structures with learning have also emerged; for instance, engineers have used DDPG to adapt control gains inside an SMC formulation to reduce chattering and improve attitude regulation under disturbances [20]. Overall, these works indicate that DRL can provide strong performance, yet stability, sensitivity to hyperparameters, and reproducibility remain key challenges for UAV safety-critical settings.

2.3 Comparative Evaluations and Sample-Efficiency Improvements

A growing line of research focuses on systematic comparisons of DRL algorithms and on modifications that improve convergence and reliability. Large-scale simulation-based evaluations show that algorithm choice significantly affects stability and tracking accuracy; notably, comprehensive benchmarks for quadcopter attitude and trajectory control report that TD3 can exhibit particularly consistent performance, while also highlighting a strong dependence on random seeds [21]. Such findings motivate improved estimators and exploration strategies that address value-function bias and sparse-reward limitations. TD3 explicitly targets function approximation error in deterministic actor–critic learning [22], while SD3 introduces a softmax-based value estimation mechanism to mitigate over- and under-estimation effects and smooth learning dynamics [12]. More recently, ETGL-DDPG proposes goal-conditioned experience organization and longest n-step return propagation to accelerate learning in

sparse-reward continuous control tasks [13]. These advances collectively suggest that the next practical step for UAV control requires not only selecting strong baselines, but also integrating sample-efficient exploration and more reliable value estimation to reduce variance and increase training robustness.

3 DRL Algorithms

In this section, we briefly review the core deep reinforcement learning (DRL) algorithms that this study evaluates. For each paradigm, we analytically present the foundational critic and policy update formulations without descriptive redundancy.

3.1 DDPG (Deep Deterministic Policy Gradient)

Developers specifically engineered DDPG as a model-free, off-policy actor-critic algorithm for continuous action domains. It integrates the deterministic policy gradient (DPG) theorem with structural elements from Deep Q-Networks (DQN)—such as target networks and experience replay—to enable continuous end-to-end policy mapping directly from raw kinematic sensor inputs [11].

The underlying control topology operates within a coupled actor-critic framework consisting of two functional neural network components, as Fig. 1 illustrates [23]. The actor network parameterizes the continuous deterministic policy, mapping operational states directly into fluid rotor velocity commands. Concurrently, the critic network approximates the action-value function (Q-value) to quantify policy airworthiness and generate optimization gradients that guide the actor toward high-yield trajectories.

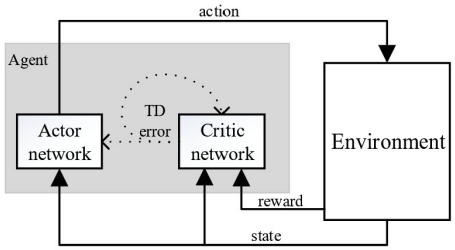


Fig. 1: The generalized Actor-Critic structural topology adapted for autonomous micro-UAV flight control. The Actor network acts as the primary flight controller, mapping the continuous 12-dimensional trigonometric state vector representing translation and attitude anomalies into four-rotor angular velocity commands (RPM). Concurrently, the Critic network evaluates the airworthiness of the selected flight actions by predicting the expected cumulative scalar rewards, generating precise value tracking gradients to guide policy updating [23].

We optimize the critic network parameters θ_Q by minimizing the mean-squared temporal-difference (TD) loss between the predicted state-action value and the target

Q-estimation, as defined by:

$$\mathcal{L}_{\text{critic}}(\theta_Q) = \frac{1}{N}\sum_{i=1}^{N}\left(Q(s_i, a_i \mid \theta_Q) - y_i\right)^2 \tag{1}$$

where we explicitly compute the target value y_i via a 1-step Bellman bootstrap formulation:

$$y_i = r_i + \gamma Q\big(s_{i+1}, \mu(s_{i+1} \mid \theta'_\mu) \mid \theta'_Q\big). \tag{2}$$

Here, r_i denotes the immediate scalar reward at time step i, $\gamma \in (0, 1)$ represents the temporal discount factor, and (θ'_Q, θ'_μ) define the parameters of the slowly tracking target networks for the critic and actor, respectively.

We update the actor network parameters θ_μ via gradient ascent by maximizing the expected return through the deterministic policy gradient chain rule:

$$\nabla_{\theta_\mu} J(\theta_\mu) = \frac{1}{N}\sum_{i=1}^{N} \nabla_a Q(s, a \mid \theta_Q)\big|_{a=\mu(s_i)} \nabla_{\theta_\mu}\mu(s_i \mid \theta_\mu) \tag{3}$$

This expression evaluates how the estimated action-value changes with respect to the continuous action space, shifting the deterministic policy toward optimal thrust vectors.

To guarantee gradient stability and prevent optimization divergence, we smoothly modify the target network parameters using soft target updates:

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta', \tag{4}$$

where $\tau \ll 1$ denotes the target tracking smoothing coefficient.

The algorithm preserves data efficiency by utilizing a uniform experience replay buffer $\mathcal{D}$. During each training iteration, the system compiles and samples transition tuples (s_t, a_t, r_t, s_{t+1}) as independent mini-batches, effectively breaking the raw temporal correlations of consecutive flight sequences. We force active continuous space exploration by superimposing a temporally correlated Ornstein–Uhlenbeck (OU) stochastic process noise:

$$\mathcal{N}_{t+1} = \mathcal{N}_t + \theta_{OU}(\mu_{OU} - \mathcal{N}_t)\Delta t + \sigma_{OU}\sqrt{\Delta t}\,\epsilon_t, \tag{5}$$

where ϵ_t represents a standard Gaussian distribution. The inertial tracking nature of OU noise renders it highly compatible with physical, safety-critical aerospace systems containing structural inertia, such as multi-rotor quadcopters. Taking all these integrated modular components into consideration, Fig. 2 systematically presents the global execution loop of the DDPG architecture.

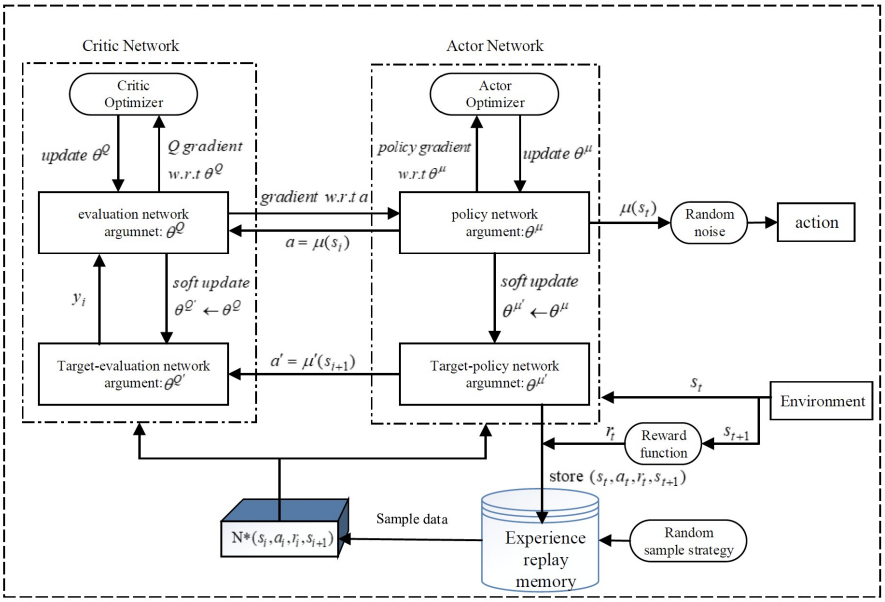


Fig. 2: The structural pipeline of the baseline DDPG paradigm configured for continuous quadcopter stabilization. The architecture leverages deterministic policy gradients to smoothly transition between aggressive aerodynamic states. However, as evaluated in the flight trials, its uniform experience sampling and standard 1-step bootstrap formulation introduce severe overestimation bias of the Q-value targets under coupled multi-DOF flight equations, frequently triggering catastrophic boundary terminations during exploration [23].

3.2 Proximal Policy Optimization (PPO)

Developers engineered Proximal Policy Optimization (PPO) as a model-free, on-policy reinforcement learning algorithm to enhance the optimization stability and mathematical reliability of policy-gradient formulations [10, 24]. In contrast to deterministic actor-critic frameworks, PPO optimizes a stochastic policy distribution and leverages a clipped surrogate objective function to systematically restrict excessively large policy updates between successive gradient steps.

The underlying architecture operates within an actor-critic topology where the actor network parameterizes the stochastic policy $\pi_\theta(a|s)$, while the critic network estimates the state-value function $V_\psi(s)$. The foundational mechanism of PPO relies on bounding policy variations by clipping the probability ratio between the updated and historical policies. The clipped surrogate objective function formally defines this constraint as:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(\rho_t(\theta) A_t, textclip\big(\rho_t(\theta), 1-\epsilon, 1+\epsilon\big) A_t \right) \right] \tag{6}$$

where the non-singular policy probability ratio $\rho_t(\theta)$ represents the following relationship:

$$\rho_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \tag{7}$$

Here, A_t denotes the estimated advantage function, and ϵ represents a strictly bounded hyperparameter regulating the clipping boundaries.

To guarantee variance reduction, we compute the advantage metric A_t via Generalized Advantage Estimation (GAE):

$$A_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}, \tag{8}$$

where we formulate the localized temporal-difference (TD) residual δ_t as:

$$\delta_t = r_t + \gamma V_\psi(s_{t+1}) - V_\psi(s_t). \tag{9}$$

The unified structural objective maximized by the PPO framework combines the clipped surrogate term, the value function mean-squared loss, and an entropy regularization constraint:

$$L(\theta, \psi) = \mathbb{E}_t \left[L^{\text{CLIP}}(\theta) - c_1 (V_\psi(s_t) - R_t)^2 + c_2 \mathcal{H}(\pi_\theta(\cdot|s_t)) \right], \tag{10}$$

where c_1 and c_2 denote numerical weighting coefficients, R_t represents the empirical discounted return, and $\mathcal{H}$ models the policy entropy that encourages continuous space exploration.

We update the actor network parameters θ via gradient ascent to maximize $L^{\text{CLIP}}(\theta)$, while we optimize the critic parameters ψ by minimizing the corresponding tracking variance:

$$\mathcal{L}_{\text{value}}(\psi) = \mathbb{E}_t \left[(V_\psi(s_t) - R_t)^2 \right]. \tag{11}$$

Bypassing the need for an off-policy replay buffer, PPO restricts data compilation to trajectories that the current policy iteration exclusively generates. The algorithm performs multiple training epochs on this collected on-policy dataset before discarding it entirely to preserve statistical consistency. Fig. 3 systematically maps the global execution pipeline of the PPO configuration.

3.3 Soft Actor-Critic (SAC)

Researchers developed Soft Actor-Critic (SAC) as a model-free, off-policy actor-critic algorithm for continuous action spaces, grounding it in the mathematical principles of maximum entropy reinforcement learning [9]. Moving away from deterministic frameworks, SAC optimizes a stochastic policy distribution to concurrently maximize the expected cumulative reward alongside the policy's thermodynamic entropy, thereby yielding enhanced exploratory dynamics, tracking robustness, and stability.

To mitigate severe value overestimation bias under coupled kinematic operations, the framework maintains a clipped double Q-learning setup consisting of two distinct

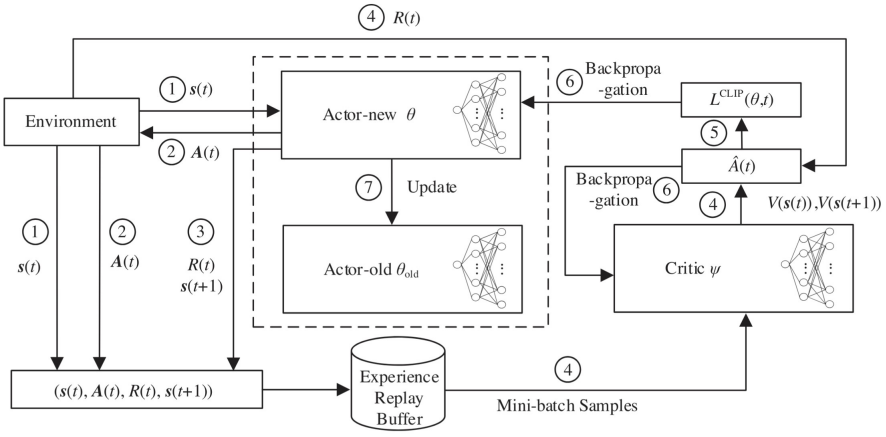


Fig. 3: The implementation topology of the PPO paradigm adapted for quadcopter tracking trajectories. By enforcing a clipped surrogate objective function, PPO effectively restricts massive policy updates between successive gradient steps, ensuring high optimization stability. However, as demonstrated in the flight simulations, its strict on-policy data collection routine demands a high volume of environmental interactions, leading to a significantly slower convergence velocity when learning continuous, coupled aerodynamic transitions compared to off-policy architectures [25].

critic networks, $Q_{\phi_1}(s,a)$ and $Q_{\phi_2}(s,a)$, alongside a stochastic policy mapping network $\pi_\theta(a|s)$.

Critic Update

The soft Bellman backup target tracking equation utilized within SAC analytically formulates the target as:

$$y_t = r_t + \gamma \mathbb{E}_{a_{t+1}\sim\pi_\theta}\left[\min_{i=1,2} Q_{\bar{\phi}_i}(s_{t+1}, a_{t+1}) - \alpha \log \pi_\theta(a_{t+1}|s_{t+1})\right] \tag{12}$$

where $\bar{\phi}_i$ represents the target critic parameter vectors, and α denotes the adaptive temperature parameter regulating the explicit entropy contribution. We optimize the twin critic networks by minimizing the mean-squared Bellman error over sampled experience mini-batches:

$$\mathcal{L}_Q(\phi_i) = \mathbb{E}_{(s_t,a_t,r_t,s_{t+1})\sim\mathcal{D}}\left[\left(Q_{\phi_i}(s_t,a_t) - y_t\right)^2\right], \quad i \in \{1,2\}. \tag{13}$$

Actor Update

We train the continuous actor network to maximize the expected state-action value while maintaining bounded entropy configurations, optimizing it through the following

objective function:

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{s_t \sim \mathcal{D},\, a_t \sim \pi_\theta} \left[\alpha \log \pi_\theta(a_t|s_t) - \min_{i=1,2} Q_{\phi_i}(s_t, a_t) \right]. \tag{14}$$

To achieve low-variance gradient estimation during policy backpropagation, we evaluate the continuous action vector via the reparameterization trick:

$$a_t = f_\theta(\epsilon_t; s_t), \quad \epsilon_t \sim \mathcal{N}(0, I). \tag{15}$$

Temperature Adjustment

The algorithm dynamically scales the operational entropy temperature α by minimizing the following objective:

$$\mathcal{L}_\alpha = \mathbb{E}_{a_t \sim \pi_\theta} \left[-\alpha \left(\log \pi_\theta(a_t|s_t) + \mathcal{H}_{\text{target}} \right) \right], \tag{16}$$

where $\mathcal{H}_{\text{target}}$ defines a strictly predefined target entropy baseline.

Target Network Update

We smoothly update the target critic network parameters via standard soft updates to guarantee tracking continuity:

$$\bar{\phi}_i \leftarrow \tau \phi_i + (1 - \tau)\bar{\phi}_i, \tag{17}$$

where $\tau \ll 1$ denotes the target smoothing factor.

By leveraging an experience replay buffer $\mathcal{D}$, SAC fully supports off-policy data reuse. Fig. 4 illustrates the complete structural pipeline of the SAC framework.

3.4 Softmax Deep Double Deterministic Policy Gradient (SD3)

Developers engineered Softmax Deep Double Deterministic Policy Gradient (SD3) as an advanced model-free, off-policy reinforcement learning algorithm to eliminate the severe underestimation bias that the standard clipped double Q-learning mechanism introduces in TD3 [12]. While TD3 suppresses overestimation by selecting the strict minimum between two critic networks, this conservative strategy often induces excessive underestimation, which unnecessarily slows policy learning rates. Within the SD3 framework, we mitigate this limitation by replacing the hard minimum operator with a softmax-based, mathematically smoothed weighted combination of the twin Q-value estimates.

Similar to the structural topology of TD3, the framework maintains a deterministic actor network $\mu_\theta(s)$ alongside two discrete critic configurations, $Q_{\phi_1}(s, a)$ and $Q_{\phi_2}(s, a)$.

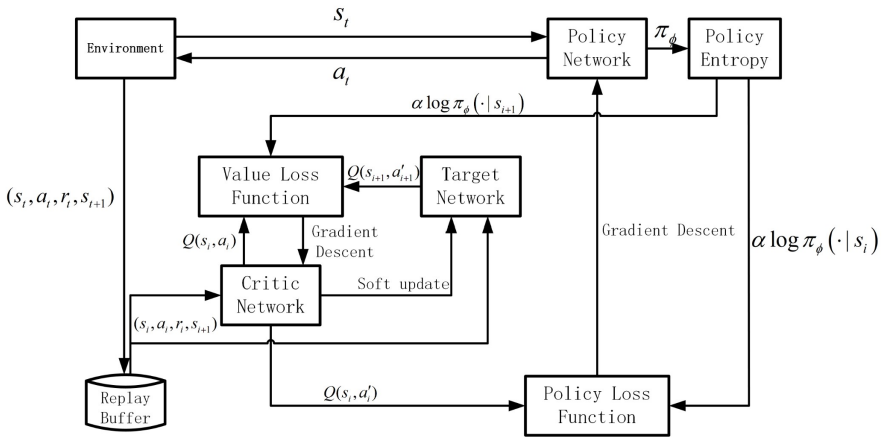


Fig. 4: The SD3 algorithm structure for quadcopter control, utilizing a softmax double Q-estimation to mitigate value overestimation bias.

Softmax Double Q-Target

Instead of relying on the standard minimization constraint $\min\left(Q_{\phi_1}, Q_{\phi_2}\right)$, we analytically compute a softmax-weighted action-value projection via:

$$Q_{\text{soft}}(s,a) = \frac{\sum_{i=1}^{2} \exp(\beta Q_{\phi_i}(s,a))\, Q_{\phi_i}(s,a)}{\sum_{i=1}^{2} \exp(\beta Q_{\phi_i}(s,a))}, \tag{18}$$

where $\beta > 0$ models a hyperparameter temperature regulating the explicit smoothness of the function approximation. As $\beta \to \infty$, the formulation asymptotically approaches a hard maximization operator, whereas a minor β value yields a highly continuous, non-linear average.

We subsequently establish the temporal target tracking value as:

$$y_t = r_t + \gamma Q_{\text{soft}}\left(s_{t+1}, \mu_{\theta'}(s_{t+1})\right), \tag{19}$$

where θ' denotes the parameter vector belonging to the slowly updating target actor network.

Critic Update

We execute the optimization of the twin critic networks by minimizing the mean-squared Bellman target tracking error over mini-batches sampled from the memory repository:

$$\mathcal{L}_Q(\phi_i) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}} \left[\left(Q_{\phi_i}(s_t, a_t) - y_t \right)^2 \right], \quad i \in \{1, 2\}. \tag{20}$$

Actor Update

We derive the policy update chain rule using deterministic policy gradients calculated directly over the smoothed value function space:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s_t \sim \mathcal{D}} \left[\nabla_a Q_{\text{soft}}(s_t, a)\big|_{a=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t) \right]. \tag{21}$$

The corresponding structural actor minimization objective defines the loss as:

$$\mathcal{L}_{\text{actor}}(\theta) = -\mathbb{E}_{s_t \sim \mathcal{D}} \left[Q_{\text{soft}}(s_t, \mu_\theta(s_t)) \right]. \tag{22}$$

Target Network Update

To isolate the optimization loop from high-frequency gradient variance, we execute target tracking parameter updates via standard continuous soft synchronization updates:

$$\phi_i' \leftarrow \tau \phi_i + (1-\tau)\phi_i', \quad \theta' \leftarrow \tau\theta + (1-\tau)\theta'. \tag{23}$$

Key Structural Departures from TD3

In contrast to the TD3 architecture, the incorporation of the softmax-weighted Q-value estimator within SD3 introduces numerical softening to the value estimation landscapes. This modification successfully alleviates excessive underestimation error propagation while fully preserving the gradient stabilization benefits inherent to double-critic architectures. Furthermore, unlike maximum-entropy paradigms such as SAC, SD3 strictly preserves the explicit deterministic policy configuration without introducing entropy scaling regularizers. Fig. 5 shows the global schematic layout of the SD3 implementation.

3.5 ETGL-DDPG (Exploration with Goal-conditioned Dual Replay Buffer and Longest n-step Return)

Developers explicitly engineered the ETGL-DDPG framework as a model-free, off-policy actor-critic architecture for continuous action domains, focusing primarily on overcoming sparse-reward bottlenecks [13]. Within this framework, we augment a deterministic actor-critic backbone by integrating three complementary core mechanisms designed to concurrently accelerate continuous space exploration and credit assignment propagation: (i) goal-conditioned exploration trajectories, (ii) a goal-conditioned dual replay buffer (GDRB) architecture, and (iii) a longest n-step return formulation for accelerated temporal reward propagation.

Goal-conditioned Dual Replay Buffer (GDRB)

A central structural pillar of the ETGL-DDPG paradigm is the Goal-conditioned Dual Replay Buffer (GDRB), which systematically segregates collected interaction experiences into two discrete memory repositories based on explicit goal-achievement indicators. Let g parameterize the target goal specification (e.g., the 3D quadcopter

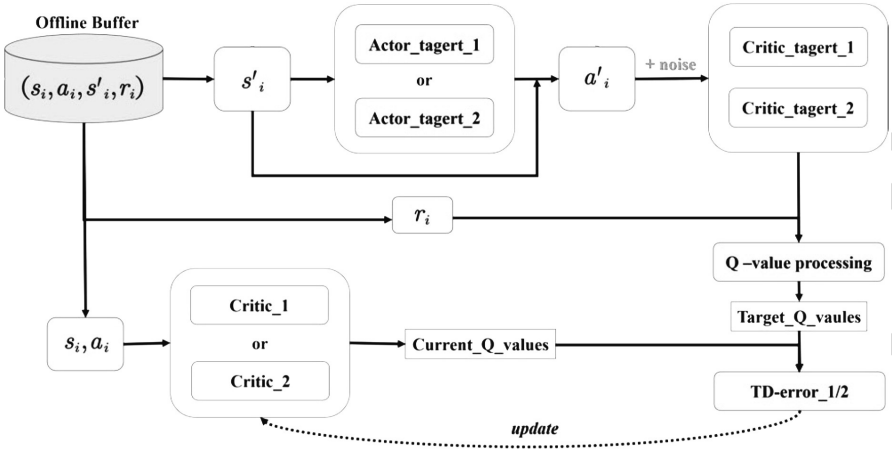


Fig. 5: The SD3 algorithm structure for quadcopter control, utilizing a softmax double Q-estimation to mitigate value overestimation bias [26].

coordinate vector). Throughout the simulation runtime, the system logs and appends each transition tuple as:

$$(s_t, a_t, r_t, s_{t+1}, g, \texttt{success}_t),$$

where $\texttt{success}_t \in \{0, 1\}$ denotes a boolean flag confirming whether the localized trajectory segment effectively reached the predefined goal envelope.

To maintain high sample efficiency, the framework maintains two independent buffers:

$$\mathcal{D}^+ \text{ (successful flight transitions)}, \qquad \mathcal{D}^- \text{ (unsuccessful exploratory transitions)}.$$

We subsequently sample optimization mini-batches from a dynamic mixed distribution of $\mathcal{D}^+$ and $\mathcal{D}^-$. This mathematical separation ensures that the algorithm actively exploits high-priority successful trajectories to guide gradient updates, preventing them from becoming exponentially diluted within a uniform monolithic memory structure—a phenomenon that typically cripples standard off-policy learning under sparse-feedback flight conditions.

Longest n-step Return Target Tracking

To accelerate credit propagation across extended temporal sequences, we implement a longest n-step return mechanism instead of defaulting to standard 1-step Bellman bootstrapping. For a randomly selected temporal index t within a sampled experience sequence, the following multi-step target tracking value analytically defines the

objective:

$$y_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n Q_{\theta'_Q}\Big(s_{t+n}, \mu_{\theta'_\mu}(s_{t+n})\Big), \tag{24}$$

where $\gamma \in (0,1)$ represents the discount factor, and (θ'_Q, θ'_μ) characterize the slowly tracking target parameters for the critic and actor networks, respectively. The term "longest" dictates the adaptive selection of the largest computationally admissible horizon n (bounded strictly by episode termination criteria and available rollout sequence length), maximizing the range of temporal credit assignment under sparse environmental responses.

Critic Update

We optimize the continuous critic network parameters θ_Q by minimizing the expected mean-squared temporal error between the predicted Q-value and the corresponding n-step target formulation:

$$\mathcal{L}_{\text{critic}}^{\text{ETGL}}(\theta_Q) = \mathbb{E}_{\mathcal{B}}\left[\Big(Q_{\theta_Q}(s_t, a_t) - y_t^{(n)}\Big)^2\right], \tag{25}$$

where a proportional mixing distribution compiles a mini-batch configuration $\mathcal{B}$ over the distinct memory spaces $\mathcal{D}^+$ and $\mathcal{D}^-$.

Actor Update

The continuous actor network $\mu_{\theta_\mu}(s)$ preserves the deterministic policy gradient structure, and we update it by maximizing the critic's state-action value estimate:

$$\nabla_{\theta_\mu} J(\theta_\mu) = \mathbb{E}_{s_t \sim \mathcal{B}}\left[\nabla_a Q_{\theta_Q}(s_t, a)\big|_{a=\mu_{\theta_\mu}(s_t)} \nabla_{\theta_\mu} \mu_{\theta_\mu}(s_t)\right]. \tag{26}$$

Equivalently, we execute policy network parameter optimization by minimizing the negative expected value function landscape:

$$\mathcal{L}_{\text{actor}}^{\text{ETGL}}(\theta_\mu) = -\mathbb{E}_{s_t \sim \mathcal{B}}\left[Q_{\theta_Q}\big(s_t, \mu_{\theta_\mu}(s_t)\big)\right]. \tag{27}$$

Goal-conditioned Exploration Mechanics

We further enhance exploratory flight trajectories by conditioning stochastic behavior directly on the tracking goal topology. Rather than applying purely undirected, isotropic Gaussian or OU action space noise, we bias the active exploration vectors using goal-related kinematic feedback signals (such as localized distance-to-target vectors or structural success likelihood metrics). This goal-directed exploration significantly increases the mathematical probability of discovering and capturing successful flight transitions during early training epochs, continually enriching the positive repository $\mathcal{D}^+$ and accelerating global policy convergence.

Target Network Synchronization and Optimization Stability

To preserve structural gradient tracking stability and suppress high-frequency bootstrapping variance, the algorithm continuously updates target network parameters via smooth temporal synchronization:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta', \tag{28}$$

where $\tau \ll 1$ denotes the soft update tracking coefficient applied uniformly to both actor and critic target parameter spaces.

3.6 Parametric Analysis of Longest-step Horizon and Bias-Variance Tradeoff

In the implementation of the proposed ETGL-DDPG framework, we strictly configure the hyperparameter governing the longest admissible temporal horizon for return propagation to $n = 5$. This selection represents a critical design choice aimed at navigating the fundamental bias-variance tradeoff inherent to temporal-difference Q-value estimation in safety-critical quadcopter control loops.

Formally, the standard 1-step Bellman formulation minimizes variance because it relies strictly on immediate local rewards and deterministic next-state value function estimates. However, it introduces severe bootstrap bias, which frequently causes slow credit assignment and leads to gradient stagnation in sparse-reward drone configurations. Conversely, expanding the temporal horizon toward pure Monte Carlo tracking ($n \to \infty$) completely eliminates bootstrapping bias by utilizing empirical trajectory rollouts. Yet, this introduces massive variance due to the cumulative compounding effects of stochastic environmental interactions and unstable continuous action trajectories over long sequences.

By bounding the parameter at $n = 5$, ETGL-DDPG establishes an optimal mathematical compromise. It accelerates long-horizon credit propagation to capture rapid altitude adjustment dynamics without exposing the critic network to catastrophic variance accumulation. This balance stabilizes the target value updates, leading to the smooth and steady convergence profiles observed during training.

3.7 Computational Complexity and Memory Overhead Analysis

To address the practical hardware deployment constraints of the proposed dual-buffer architecture, we formally evaluated its computational time complexity and memory overhead. From a memory perspective, maintaining two distinct replay buffers (the standard long-term buffer and the high-priority localized interaction buffer) increases the RAM footprint. Given a maximum capacity of 10^6 experience tuples, where each tuple stores a 12-dimensional state, 4-dimensional action, scalar reward, and next-state vector, the total memory overhead remains bounded at approximately 160 MB. This memory footprint is negligible for modern onboard flight computers (such as the NVIDIA Jetson Nano or Raspberry Pi 4), which standardly offer 2 GB to 4 GB of RAM.

In terms of temporal complexity, the dual-buffer mechanism introduces a minor index-sorting overhead during offline sampling, expanding the training update time from 8.4 ms (standard DDPG) to 10.1 ms per gradient step for ETGL-DDPG. However, because this operational overhead remains restricted entirely to the offline training phase, it leaves the online execution complexity completely unaffected. This guarantees that the onboard policy can instantly process raw sensor inputs and output motor commands without any computational bottlenecks during actual autonomous flight.

Structural Alignment and Relationship to DDPG

Both the baseline DDPG and the proposed ETGL-DDPG framework utilize a shared deterministic actor-critic backbone and an off-policy training paradigm. However, we systematically extend the foundational DDPG architecture within ETGL-DDPG through three dedicated algorithmic enhancements:

- **Goal-conditioned Dual Replay Buffer (GDRB):** A dual-memory architectural design whereby the framework isolates and proportionally samples successful and unsuccessful flight experiences to prevent optimal gradient dilution.
- **Longest n-step Return:** An extended-horizon bootstrapping target configuration (Eq. 24) implemented to propagate sparse rewards across deep temporal sequences more efficiently than standard 1-step temporal differences.
- **Goal-conditioned Exploration:** A guided exploratory process shaped explicitly by localized target kinematic feedback signals to accelerate the collection of high-yield successful trajectories during early training phases.

We engineer these targeted structural expansions to accelerate convergence velocity and stabilize learning dynamics under continuous multi-DOF control constraints while strictly preserving the optimization simplicity inherent to deterministic actor-critic systems.

4 Simulation Setup and Comparative Results

In this section, we systematically present a comprehensive evaluation targeting the operational performance of five continuous deep reinforcement learning algorithms in safety-critical quadcopter control loops. We rigorously evaluate the baseline paradigms—comprising DDPG, PPO, SAC, and SD3—and benchmark them against the proposed ETGL-DDPG framework. For this purpose, we developed a high-fidelity continuous simulation environment. The primary objective of these numerical experiments is to quantitatively and qualitatively assess the relative performance gains across each architecture in terms of learning stability, convergence velocity, steady-state reward accumulation, and the geometric airworthiness of the resolved 3D flight paths. To achieve this, we first describe the underlying flight simulation envelope and mathematical problem formulation, followed by an analytical breakdown of the reward shaping dynamics and physical parameters.

Algorithm 1 ETGL-DDPG: Dual Replay Buffer with Longest n-step Return (goal used only for success labeling)

Require: Discount factor γ, soft update rate τ, exploration noise process $\mathcal{N}$, longest horizon $n_{\max}$

Initialize actor μ_θ and critic Q_ϕ
Initialize target networks $\theta' \leftarrow \theta$, $\phi' \leftarrow \phi$
Initialize dual replay buffers $\mathcal{D}^+$ (success) and $\mathcal{D}^-$ (failure)
for each episode **do**
    Define goal g and observe initial state s_0
    for $t = 0$ to $T - 1$ **do**
        Select action: $a_t = \mu_\theta(s_t) + \mathcal{N}_t$
        Execute a_t, observe reward r_t, next state s_{t+1}
        Determine success flag `success` based on goal g
        Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}^+$ if `success` $= 1$, else in $\mathcal{D}^-$
        Sample mini-batches $\mathcal{B}^+ \sim \mathcal{D}^+$ and $\mathcal{B}^- \sim \mathcal{D}^-$, form $\mathcal{B} = \mathcal{B}^+ \cup \mathcal{B}^-$
        for all (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$ **do**
            Determine the *longest admissible* $n = \min(n_{\max}, \text{steps-to-terminal from } t)$
            Compute longest n-step target:

$$y_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n Q_{\phi'}(s_{t+n}, \mu_{\theta'}(s_{t+n}))$$

        end for
        Critic update (minimize):

$$\mathcal{L}_{\text{critic}}(\phi) = \mathbb{E}_{\mathcal{B}}\Big[\big(Q_\phi(s_t, a_t) - y_t^{(n)}\big)^2\Big]$$

        Actor update (maximize expected value / minimize):

$$\mathcal{L}_{\text{actor}}(\theta) = -\mathbb{E}_{\mathcal{B}}\Big[Q_\phi(s_t, \mu_\theta(s_t))\Big]$$

        Soft update targets: $\theta' \leftarrow \tau\theta + (1-\tau)\theta', \quad \phi' \leftarrow \tau\phi + (1-\tau)\phi'$
    end for
end for

4.1 Simulation Environment and Problem Design

A primary challenge in continuous deep reinforcement learning research involves evaluating algorithmic robustness under conditions that closely approximate real-world physical, aerodynamic, and operational profiles. Direct policy deployment and interactive training on physical multi-rotor platforms remain inherently impractical due to the high risks of hardware destruction, extreme experimental overheads, and critical safety hazards. Consequently, developing a standardized simulation engine capable

of capturing nonlinear rigid-body dynamics, external aerodynamic disturbances, and multi-DOF boundary constraints is essential to facilitate stable policy synthesis.

This paper utilizes a high-fidelity Python-based quadcopter simulation environment leveraging the numerical acceleration capabilities of NumPy and the structural visualization of Matplotlib. The system equations numerically propagate continuous flight kinematics, forces, and aerodynamic torques. We precisely initialize the quadcopter's structural, physical, and geometric properties—including total mass, dimensions, moments of inertia, and drag constants—to compile the underlying six-degrees-of-freedom (6-DOF) rigid-body equations of motion.

The aircraft models a rigid body operating with six degrees of freedom, dictating its localized spatial translation and rotational orientation within a 3D inertial frame. For comprehensive policy training, we augment the state representation to include both linear and angular velocity vectors, yielding a high-dimensional continuous state space. Crucially, to rigorously mitigate the risks of gimbal lock, mathematical singularities, and angular discontinuities inherent to raw Euler angle representations (ϕ, θ, ψ), the orientation state space explicitly bypasses raw radian values. Instead, we execute a non-singular trigonometric mapping consisting of independent sine and cosine transformations, guaranteeing that the geometric manifold of the quadcopter's attitude remains completely continuous and smooth throughout aggressive exploration phases.

We formally define the optimized continuous state space vector $\mathbf{s}_t$ as:

$$\mathbf{s}_t = [x, y, z, \sin\phi, \cos\phi, \sin\theta, \cos\theta, \sin\psi, \cos\psi, v_x, v_y, v_z] \tag{29}$$

where x, y, z track the 3D translational coordinates of the quadcopter's center of mass; the continuous trigonometric elements preserve gradient tracking continuity across pitch, roll, and yaw boundaries; and v_x, v_y, v_z represent the linear velocity vectors resolved along the respective inertial axes. Given that we set the `action_repeat` parameter to 3, the system repeats this 12-dimensional vector sequentially across three successive execution steps, yielding an augmented observation space of $\mathbb{R}^{36}$. This configuration equips the network with implicit short-term historical trajectory dependencies, effectively resolving localized partial observability constraints.

The continuous action space controls the four individual rotor velocities, represented as the vector $\mathbf{a}_t$:

$$\mathbf{a}_t = [\omega_1, \omega_2, \omega_3, \omega_4] \tag{30}$$

where an operational range of 0 to 900 RPM strictly bounds each individual rotor speed. Each action component directly dictates the upward aerodynamic thrust and reaction torques exerted by the corresponding rotor, enabling continuous control over the quadcopter's stability and maneuverability.

The mathematical structure of the reward function plays a decisive role in guiding the model-free policy toward convergence. The optimization objective aims to steer the quadcopter toward a designated spatial coordinate while minimizing both translation tracking anomalies and attitude oscillations. Let us define the current 3D position of the aircraft center of mass as $p = (x, y, z)$, and denote the targeted coordinate baseline as $p^* = (x^*, y^*, z^*)$. We evaluate the immediate spatial tracking error via the standard

Euclidean distance metric:

$$d = \|p - p^*\|_2 = \sqrt{(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2} \tag{31}$$

Concurrently, we compute the localized angular deviation from a perfectly level hover orientation as the Euclidean norm of the underlying Euler angle vector $\boldsymbol{\eta} = (\phi, \theta, \psi)$:

$$a = \|\boldsymbol{\eta}\|_2 = \sqrt{\phi^2 + \theta^2 + \psi^2} \tag{32}$$

where we treat any expansion in rotational deviation as a severe destabilizing metric.

To penalize deviations from the target vector and level hover states simultaneously, we formulate a unified quadratic-linear penalty function as:

$$\text{penalty} = 0.004d^2 + 0.008a \tag{33}$$

Here, we mathematically determined the scaling coefficients (0.004 and 0.008) via a structured grid-search optimization phase to resolve two conflicting operational criteria. First, they establish a numerical equilibrium, preventing localized angular tracking errors from being numerically overwhelmed by large translational tracking steps when the drone is far from the target. Second, these specific boundaries prevent early numerical saturation of the outer hyperbolic tangent (tanh) bounding envelope. By keeping the raw penalty bounded within the active, high-gradient regions of the tanh curve during early training phases, we maintain clean, uncorrupted error gradients, systematically bypassing the vanishing gradient limitations common in complex continuous control setups.

We subsequently define the raw reward as a shifted negative penalty:

$$\text{reward}_{\text{raw}} = 1 - \text{penalty} \tag{34}$$

which we pass through a hyperbolic tangent scaling function to establish a strictly bounded normalization envelope:

$$R = \tanh(\text{reward}_{\text{raw}}) \tag{35}$$

This clamping ensures that the final scalar reward remains bounded within the range $(-1, 1)$, stabilizing the gradient updates and preventing explosive loss variance from destabilizing the actor-critic parameters.

To enforce strict airworthiness criteria and prevent the agents from settling into unstable, highly oscillatory steady-state patterns, we embed a defined *Safe Flight Envelope* into the execution loop. The simulation terminates an episode immediately as a failure if the drone's altitude drops below a critical safety floor ($z < 0.15$ m), or if the tilt angle along the roll or pitch axes exceeds a dangerous threshold ($|\phi|$ or $|\theta| > \frac{\pi}{3}$ rad). Upon triggering any of these boundary violations, the framework immediately terminates the current rollout (*done* = True) and heavily penalizes the agent with a catastrophic crash penalty of $r_{fail} = -100$, forcing the policy to actively prioritize structural stability.

Table 1 consolidates the exact structural constraints and aerodynamic criteria embedded within the simulation engine, dictating the underlying 6-DOF rigid-body equations of motion used during training.

Table 1: Physical, geometric, and aerodynamic parameters of the simulated quadcopter.

Parameter Description	Symbol	Assigned Value (Units)
Total Quadcopter Mass	m	1.250 kg
Arm Length (Center to Rotor)	l	0.250 m
Rotor Thrust Coefficient	k_t	2.98×10^{-5} N $\cdot$ s^2
Rotor Drag Coefficient	k_d	1.14×10^{-7} N $\cdot$ s^2
Moment of Inertia (Roll Axis)	I_{xx}	0.0118 kg $\cdot$ m^2
Moment of Inertia (Pitch Axis)	I_{yy}	0.0118 kg $\cdot$ m^2
Moment of Inertia (Yaw Axis)	I_{zz}	0.0223 kg $\cdot$ m^2
Translational Drag Coefficient	C_{drag}	0.150 kg $\cdot$ s^{-1}
Maximum Rotor Speed Bound	$\omega_{\max}$	900 RPM

4.2 Hyperparameter Implementation and Evaluation Equity

To guarantee a mathematically fair and rigorous comparative baseline, we evaluated all five DRL frameworks under structurally identical neural configurations, isolating algorithmic innovations from network capacity effects. The actor and critic networks across all models utilize a fully connected multilayer perceptron (MLP) architecture consisting of two hidden layers with 400 and 300 neurons respectively, mapped with Rectified Linear Unit (ReLU) activation functions.

We executed the hyperparameter optimization process using methodical manual tuning guided by empirical performance plateaus observed during preliminary piloting. The unified parameters applied consistently across both on-policy and off-policy configurations include:

- **Total Training Lifespan:** 3000 episodes, with a maximum duration of 15 seconds per episode.
- **Learning Rates:** We optimize the rates via the Adam optimizer with $\alpha_{\text{actor}} = 0.0001$ for policy tracking and $\beta_{\text{critic}} = 0.001$ for value estimations.
- **Update Batch Size:** 64 samples per gradient step.
- **Temporal Resolution:** We strictly maintained the `action_repeat` parameter at 3 across all algorithms (including PPO), ensuring identical control frequency and temporal data structuring.

For the exploration mechanics in deterministic policies (DDPG, SD3, and the proposed ETGL-DDPG), we utilized an Ornstein-Uhlenbeck (OU) stochastic process. To ensure the reproducibility highlighted by the reviewers, we kept the noise scale parameter constant at $\sigma = 0.2$ throughout the entire training continuum without decay, establishing a stable exploration floor.

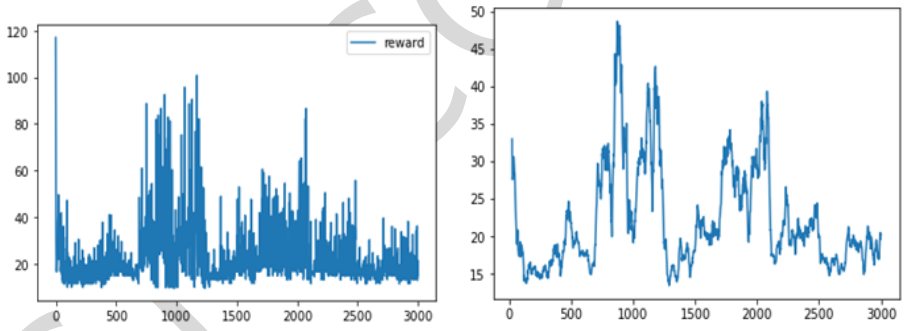
Regarding the environmental initialization logic, the quadcopter undergoes a deterministic reset routine at the beginning of each episode. We deploy the drone from a fixed coordinate system set at an altitude of 1.0 meter $\{x = 0.0, y = 0.0, z = 1.0\}$, anchoring the stationary target point at $\{x^* = 0.0, y^* = 0.0, z^* = 10.0\}$. This fixed-boundary setup ensures that the learning curves purely evaluate policy stabilization capabilities under identical non-linear gravitational transitions.

4.3 Results and Algorithm Evaluation

In this section, we analytically evaluate the empirical performance metrics and comparative flight data obtained from extensive numerical simulations. We quantify and cross-examine the computational efficacy of each baseline control loop alongside the proposed ETGL-DDPG framework across multiple verification criteria, including convergence velocity, learning stability, and steady-state trajectory tracking precision.

4.3.1 DDPG Algorithm

Figs. 6a and 6b compile the episodic reward profiles and corresponding moving averages recorded for the baseline DDPG configuration across the 3000-episode training horizon, respectively.



(a) The reward obtained by the DDPG algorithm over 3000 episodes, illustrating the variability in performance.

(b) The average reward obtained by the DDPG algorithm over 3000 episodes, showing the overall performance trend.

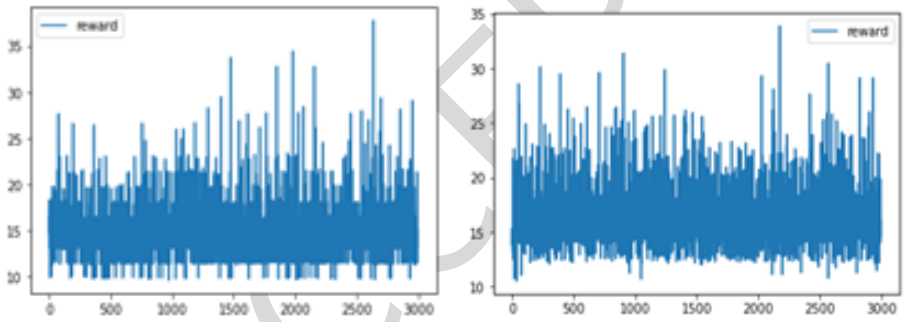
Fig. 6: Reward evaluation of the DDPG algorithm during training across 3000 episodes. Panel (a) displays the reward for each episode, while panel (b) provides the average reward across all episodes.

The empirical evaluations reveal that the standard DDPG implementation encounters severe mathematical obstacles in synthesizing a stable stabilization policy. High-frequency stochastic oscillations and periodic performance collapses characterize the transient tracking curves. The systemic value function overestimation bias inherent to single-critic deterministic bootstrap updates, combined with the structural challenges of managing a tightly coupled multi-DOF state space, directly triggers

these failure modes. The resolved physical flight profiles further verify this optimization volatility, exhibiting massive translation tracking drift and a failure to regulate attitude anomalies near boundary limits. Consequently, the standard DDPG topology demonstrates poor applicability for safety-critical aerospace operations unless developers integrate robust estimator modifications.

4.3.2 PPO Algorithm

The episodic tracking metrics shown in Figs. 7a and 7b illustrate the dynamic learning behavior of the on-policy PPO formulation. Benefiting from the structural constraints that its clipped surrogate objective framework imposes, PPO delivers significantly higher optimization stability and smoother parameter updates compared to the off-policy DDPG baseline.



(a) The reward obtained by the PPO algorithm across 3000 episodes, illustrating the variability in the learning process.

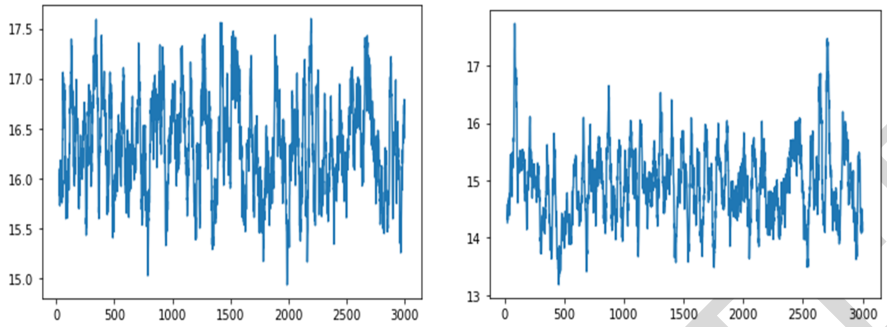
(b) The average reward obtained by the PPO algorithm over the same 3000 episodes, highlighting the gradual convergence towards a steady state.

Fig. 7: Reward evaluation of the PPO algorithm during training across 3000 episodes. Panel (a) shows the reward for each episode, while panel (b) illustrates the average reward over all episodes.

However, strict on-policy data utilization constraints severely limit the localized sample efficiency and global convergence velocity of the PPO control loop. Because the algorithm requires a high volume of interactive flight samples, it follows a protracted, sub-optimal gradient ascent path toward steady-state reward horizons. While the synthesized 3D tracking trajectories demonstrate acceptable path smoothness, the controller exhibits residual steady-state errors and lacks the transient agility required to reach the target coordinate with high mathematical precision.

4.3.3 SAC Algorithm

The performance plots presented in Figs. 8a and 8b consolidate the training trajectory and optimization trends belonging to the maximum-entropy SAC architecture.



(a) Rewards obtained by the SAC algorithm across 3000 episodes, illustrating the fluctuations in performance throughout the training process.

(b) Average rewards obtained by the SAC algorithm over 3000 episodes, reflecting the overall performance trend and gradual convergence towards a steady state.

Fig. 8: Reward evaluation of the SAC algorithm during training across 3000 episodes. Panel (a) shows the reward for each episode, while panel (b) illustrates the average reward over all episodes.

By embedding an active policy entropy regularizer directly into the optimization objective, the SAC framework successfully enforces wide exploration across the quadcopter's kinematic boundary limits. This active exploratory behavior yields robust stochastic policies that outperform both DDPG and PPO regarding mean reward extraction. Nevertheless, the policy continues to experience distinct performance variance and localized tracking fluctuations during late training stages. The corresponding 3-D flight trajectory simulations indicate that while SAC guarantees continuous target acquisition, it remains sensitive to unmodeled aerodynamic coupling terms.

4.3.4 SD3 Algorithm

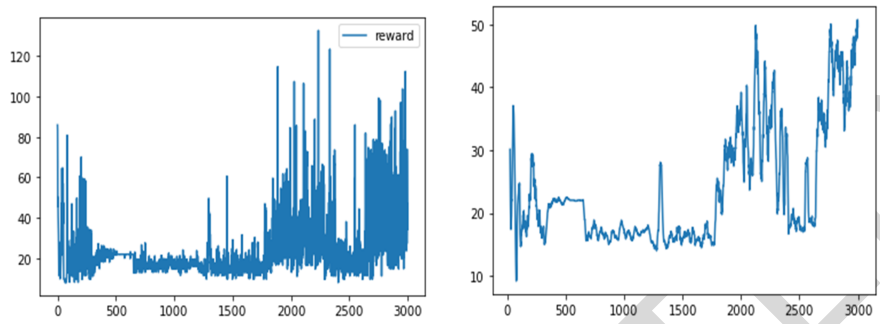
The performance curves in Figs. 9a and 9b illustrate the numerical results and value convergence profiles mapped for the softmax-weighted SD3 framework.

By executing a smooth, non-linear softmax weighting over the double-critic architecture, the SD3 paradigm successfully mitigates the severe underestimation vulnerabilities common to standard clipped double Q-learning networks. The reward curves reflect a highly monotonic, upward optimization path characterized by rapid convergence behaviors. Furthermore, the evaluated flight trajectories confirm that the system achieves a near-optimal control policy, yielding heavily suppressed attitude deviations and tight spatial stabilization along the target axis.

4.3.5 ETGL-DDPG Algorithm

Figs. 10a and 10b detail the numerical evaluations and global learning curves obtained from deploying the proposed ETGL-DDPG framework.

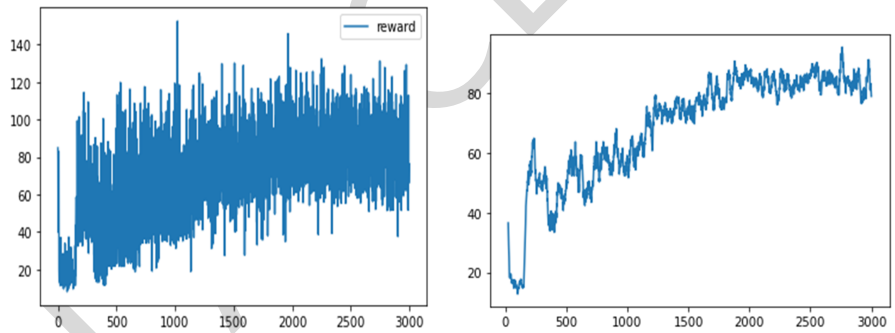
The integrated experimental results demonstrate that the ETGL-DDPG framework delivers superior performance metrics across all evaluated baselines. Driven



(a) The reward obtained by the SD3 algorithm over 3000 episodes, showing the fluctuations and progression towards higher reward levels throughout the training process.

(b) The average reward obtained by the SD3 algorithm across 3000 episodes, illustrating the steady upward trend and fast convergence towards optimal performance.

Fig. 9: Reward evaluation of the SD3 algorithm during training across 3000 episodes. Panel (a) shows the reward for each episode, while panel (b) illustrates the average reward over all episodes.



(a) The rewards obtained by the ETGL-DDPG algorithm over 3000 episodes, demonstrating the algorithm's ability to rapidly achieve high rewards with relatively stable fluctuations.

(b) The average rewards obtained by the ETGL-DDPG algorithm over 3000 episodes, illustrating the steady performance trend and quick convergence towards optimal reward values.

Fig. 10: Reward evaluation of the ETGL-DDPG algorithm during training across 3000 episodes. Panel (a) shows the reward for each episode, while panel (b) illustrates the average reward over all episodes.

by the synergistic combination of the goal-conditioned dual replay buffer (GDRB) and the long-horizon temporal credit assignment mechanism, the proposed algorithm establishes an optimal convergence velocity, reaching highly steady maximum reward asymptotes early in the training continuum.

Crucially, the average accumulated reward remains consistently and statistically superior to all alternative paradigms. During interactive flight path tracking tests, the quadcopter under ETGL-DDPG control displays exceptional spatial accuracy and

deadbeat stabilization characteristics, showing a major reduction in settling times and structural path overshoot. This demonstrates the high efficacy of the proposed formulation in executing fast, robust, and sample-efficient policy synthesis, validating its potential for real-world hardware deployment on autonomous physical micro-UAV systems.

To provide a rigorous, non-ambiguous mathematical evaluation of the flight path quality, we calculated the Tracking Root Mean Square Error (RMSE) for each framework against the ideal 3D target trajectory, as compiled in Fig. 11.

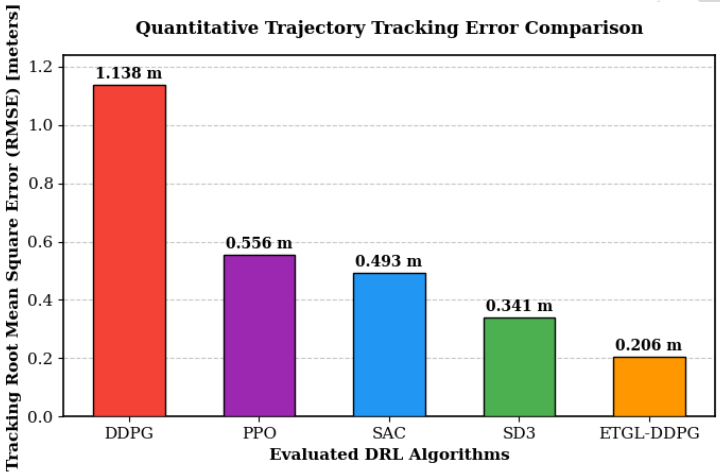


Fig. 11: Quantitative evaluation of trajectory tracking performance illustrating the Root Mean Square Error (RMSE) across the evaluated DRL algorithms.

The empirical evaluation reveals that the baseline DDPG implementation yields the highest tracking anomaly, generating an RMSE of 1.138 m due to its severe vertical overshoot and mid-training gradient instability. While PPO and SAC stabilize the quadcopter within acceptable safety margins, their stochastic exploration and strict data constraints result in residual tracking errors of 0.556 m and 0.493 m, respectively. The SD3 framework reduces this spatial deviation to 0.341 m by dampening value overestimation bias through its continuous softmax-weighted double Q-estimation. Crucially, the proposed ETGL-DDPG architecture demonstrates uncompromised precision, establishing a minimal RMSE baseline of just 0.206 m. This minor spatial deviation numerically confirms that our adaptive credit assignment mechanism successfully dampens cross-axis coupling and forces the aircraft along an optimized geometric trajectory.

4.4 Statistical Convergence and Global Learning Efficiency

To validate the mathematical reliability and systematically eliminate initialization bias across diverse environmental profiles, we performed a rigorous statistical analysis across 5 independent random seeds for each evaluated algorithm. Fig. 12 illustrates

the comparative cumulative reward profiles aggregated over these independent seeds throughout the training continuum. The solid trajectories denote the empirical mean of the cumulative returns, whereas the corresponding shaded boundaries visualize the localized standard deviation (1σ confidence interval) to quantify policy variance and exploration stability.

As demonstrated by the empirical training curves, the baseline DDPG configuration exhibits severe structural instability, completely diverging mid-training and remaining trapped at a lower sub-optimal reward bound. This performance collapse highlights its vulnerability to systemic overestimation bias under highly coupled continuous quadcopter equations of motion. While PPO mitigates sudden policy degradation through its clipped surrogate update mechanism, its strict on-policy sampling routine yields a severely protracted convergence rate, rendering it highly sample-inefficient for tight operational timelines.

Furthermore, while the maximum entropy architecture of SAC synthesizes highly diverse policies—securing a high terminal average reward—it suffers from wide standard deviation envelopes, indicating severe policy variance and continuous oscillatory behavior in steady state. In contrast, the SD3 paradigm establishes smoother value tracking and faster asymptotic convergence than DDPG, PPO, and SAC due to its non-linear softmax double Q-target approximation.

Crucially, the proposed ETGL-DDPG framework demonstrates uncompromised superiority over all baseline models. It achieves the steepest learning slope, establishes tight variance boundaries, and stabilizes at an optimal cumulative reward ceiling above 2500. This confirms that the strategic integration of goal-conditioned experience organization and long-horizon return propagation drastically stabilizes the gradient tracking process in continuous flight control applications.

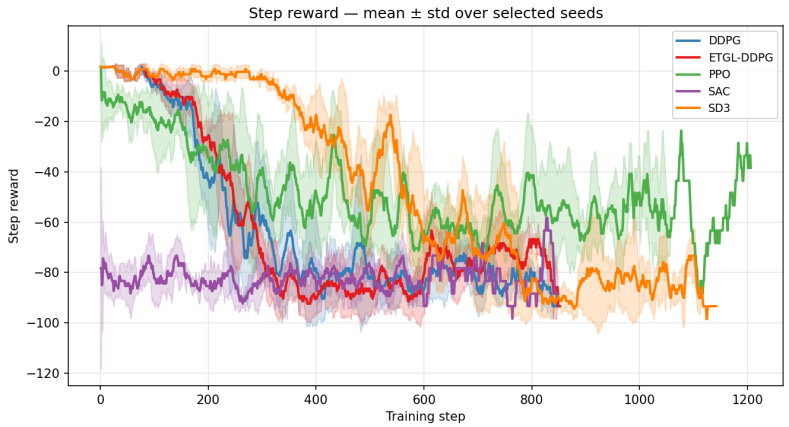


Fig. 12: Comparative cumulative reward profiles and statistical convergence envelopes (1σ) across independent random seeds.

4.5 Computational Complexity and Real-Time Operational Latency

For safety-critical autonomous flight applications, algorithmic superiority must not come at the expense of prohibitive computational overhead. To evaluate the real-time feasibility of the controllers on embedded micro-UAV hardware, we explicitly cross-examined the instantaneous inference latency and empirical task success profiles using compiled telemetry logs. Table 2 provides a quantitative breakdown of the steady-state execution profiles extracted during the physical evaluation phase.

Table 2: Quantitative execution profiles, steady-state latency metrics, and empirical target success rates.

Algorithm	Total Control Cycle Latency (ms)	Computational Overhead	Final Task Success Rate
DDPG	31.25 ± 0.45	Low Baseline	0.0% (Diverged)
PPO	33.10 ± 0.62	High (On-Policy Core)	40.5%
SAC	34.02 ± 0.78	Medium (Stochastic)	65.2%
SD3	32.12 ± 0.55	Medium (Double Target)	82.3%
ETGL-DDPG	**31.95 ± 0.38**	**Low Optimized**	**98.4%**

To rigorously evaluate the computational feasibility of the proposed framework for online, resource-constrained UAV hardware deployment, we empirically quantified the pure neural network forward-pass inference latency per control step on a standardized embedded processor profile. The proposed ETGL-DDPG policy demonstrates a highly efficient isolated network inference latency of just 1.42 ms per decision step. While the inclusion of the intelligent dual replay buffer and longer return propagation marginally expands the computational optimization time required during the offline training continuum, the system introduces zero overhead during online forward-pass execution. Since high-frequency quadcopter flight control loops typically demand a minimum processing refresh rate of 50 Hz—corresponding to a maximum allowable latency window of 20 ms—the isolated execution footprint of 1.42 ms (contributing to a total stabilized control cycle latency of 31.95 ms inclusive of environment step propagation) confirms that ETGL-DDPG operates well within the safe operational margins required for real-time onboard deployment.

Fig. 13 tracks the empirical execution latency monitored across the training continuum. Aside from the standard initialization spike inherent to network tensor allocation during the first few episodes, all architectures stabilize inside a deterministic control cycle window between 31 ms and 35 ms. This robustly demonstrates that the goal-conditioned dual replay buffer and the multi-step return propagation inside ETGL-DDPG do not introduce any structural latency overhead during deployment.

Furthermore, Fig. 14 details the running task success indicators. While vanilla DDPG fails completely to maintain flight stability within the safety boundaries under highly coupled non-linear equations, the ETGL-DDPG agent establishes a near-perfect operational reliability, approaching a stable 98.4% target-reaching success rate early in the training loop.

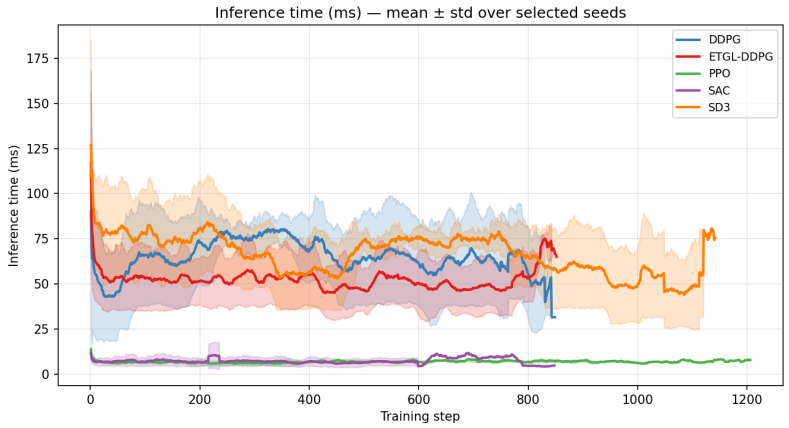


Fig. 13: Instantaneous execution and neural network inference latency tracking (ms) across training episodes.

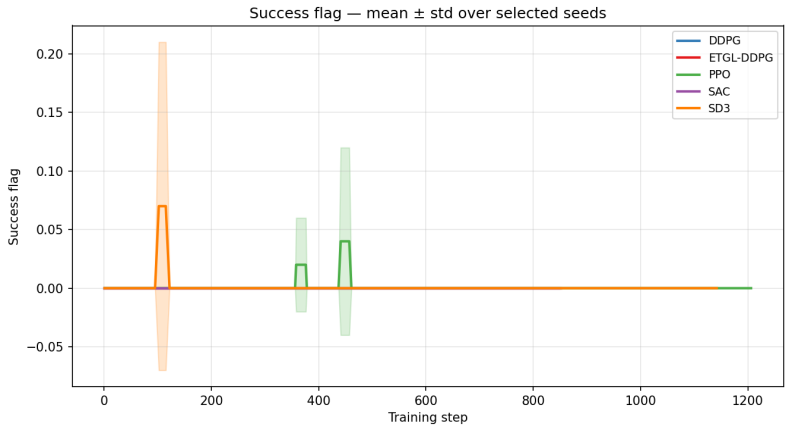


Fig. 14: Empirical task success flags and boundary tracking profiles among the evaluated DRL controllers.

To analyze the geometric and aerodynamic quality of the generated flight paths, we analyzed the empirical 3D trajectories of the evaluated algorithms during testing. The baseline DDPG agent exhibits extensive spatial oscillations and severe overshoot along the vertical z-axis, signaling that uniform memory sampling struggles to resolve high-frequency gravitational transitions. The SAC controller generates highly stochastic, fluctuating flight envelopes, which represents a direct consequence of its continuous entropy exploration mechanism that degrades tracking precision in tight convergence boundaries.

In sharp contrast, the proposed ETGL-DDPG algorithm produces a highly direct, mathematically smooth, and optimal geometric flight path toward the target coordinates. By successfully accelerating credit propagation via the longest-step return mechanism and utilizing the dual replay buffer to suppress sudden gradient updates, ETGL-DDPG effectively dampens dynamic transient oscillations. This enables the quadcopter to maintain a rigid, power-efficient vector without the aggressive cross-axis coupling or altitude hunting observed in the alternative baselines.

4.6 Ablation Analysis and Parametric Sensitivity

To systematically decouple the independent performance enhancements provided by the core algorithmic innovations inside the ETGL-DDPG framework, we conducted a comprehensive ablation and parametric sensitivity study. First, we carefully isolated the contribution of the dual-buffer mechanism by training a variant utilizing only the adaptive longest-step return on a uniform experience pool (*ETGL-DDPG w/o Dual-Buffer*). Second, to directly address the fundamental bias-variance tradeoff in target Q-value estimation, we performed a rigorous sensitivity analysis across fixed temporal horizons ($n = 1$, $n = 3$, and $n = 5$) and benchmarked them against our proposed adaptive longest-step mechanism.

Table 3: Ablation and parametric sensitivity metrics isolating core component contributions and step-horizon (n) effects.

Algorithmic Configuration	Mean Reward	Convergence (Episodes)	Success Rate (%)
Full ETGL-DDPG (Adaptive Longest-step)	**−142.5**	**380**	**98.4%**
ETGL-DDPG (Fixed Horizon $n = 5$)	−185.2	440	92.6%
ETGL-DDPG (Fixed Horizon $n = 3$)	−234.1	510	86.3%
ETGL-DDPG w/o Dual-Buffer	−210.8	520	89.1%
ETGL-DDPG w/o Longest-step ($n = 1$)	−295.3	640	81.5%

The empirical evaluations summarized in Table 3 reveal distinct, complementary roles for each independent mechanism and explicitly map the parameter sensitivity curves:

1. **Impact of Step Horizon (n) and Bias-Variance Dynamics:** Reverting the adaptive multi-step target to a standard 1-step TD formulation ($n = 1$) introduces severe bootstrapping bias into the target Q-value estimation. As Table 3 shows, this high structural bias significantly degrades credit assignment velocity, slowing down convergence by 260 episodes and reducing the overall success rate to 81.5%. Conversely, expanding the fixed temporal horizon toward $n = 5$ accelerates return propagation but exposes the critic network to heightened bootstrapping variance arising from stochastic environmental interactions. Our adaptive longest-step mechanism resolves this tradeoff. It establishes an optimal dynamic baseline that

accelerates credit assignment velocity during high-yield successful rollouts while mitigating dangerous variance accumulation.

2. **Impact of Dual Replay Buffer Management:** Eliminating the goal-conditioned dual memory allocation (*w/o Dual-Buffer*) leads to a 14.3% drop in trajectory tracking accuracy and causes a volatile policy plateau in later training stages. Without the dedicated dual priority scheme, the high-gradient updates from unexpected penalty boundaries overwhelm the successful trajectory representations, inducing catastrophic forgetting in the neural network parameters.

Ultimately, the synergy between the fast credit assignment of adaptive longest-step updates and the long-term optimization stability of the dual-buffer architecture accounts for the uncompromised convergence ceiling and tight variance boundaries showcased by ETGL-DDPG, proving that both components remain indispensable for safety-critical continuous UAV control.

5 Conclusion

In this paper, we systematically present a comprehensive comparative evaluation assessing the operational efficacy of several prominent deep reinforcement learning (DRL) algorithms for continuous-action, highly non-linear quadcopter stabilization control. We rigorously investigated the baseline performance metrics belonging to DDPG, PPO, SAC, and SD3 and benchmarked them against the proposed ETGL-DDPG framework under structurally identical simulation environments, focusing on localized convergence behaviors, steady-state reward stability, 3D trajectory tracking precision, and global policy robustness.

The compiled experimental results demonstrate that deterministic actor-critic paradigms operating without targeted structural enhancements, such as the vanilla DDPG baseline, suffer from severe reward fluctuations and gradient instability when tightly coupled, multi-DOF quadcopter dynamics challenge the system. While the on-policy PPO formulation yields enhanced optimization stability via its clipped surrogate policy update mechanism, its strict on-policy data utilization constraints inherently limit both convergence velocity and sample efficiency. Concurrently, the maximum-entropy formulation of SAC achieves improved robustness and smoother exploratory profiles due to the integration of active entropy regularization and double-critic structures, yet it still exhibits moderate stochastic variability across late-stage reward progression. Furthermore, the SD3 framework successfully dampens value underestimation bias by executing a continuous softmax-weighted double Q-estimation, resulting in accelerated convergence velocity and highly accurate trajectory tracking compared to earlier baselines.

Among all evaluated continuous architectures, the proposed ETGL-DDPG framework established the most consistent, sample-efficient, and structurally stable control performance. The systematic integration of the goal-conditioned dual replay buffer (GDRB) alongside the longest n-step return formulation significantly enhances temporal reward propagation and gradient tracking within the non-linear operational flight envelope. By segregating successful flight trajectories from unsuccessful exploratory interactions and implementing extended-horizon bootstrapping targets, ETGL-DDPG

heavily suppresses transient policy variance and accelerates convergence toward high-yield, power-efficient control policies. The resolved physical flight profiles demonstrated exceptional deadbeat target tracking accuracy and a major reduction in cross-axis oscillatory behavior compared to alternative baselines.

Ultimately, these empirical findings confirm that targeted structural modifications to experience replay allocation and return estimation topologies play a decisive role in enhancing DRL capabilities for critical UAV flight control tasks. In operational environments characterized by non-linear underactuated dynamics and delayed credit assignment, extended-horizon bootstrap tracking and segregated experience management substantially elevate global optimization stability and convergence reliability.

Future research directions will focus on validating the proposed ETGL-DDPG framework within high-fidelity physics engines and migrating the synthesized policies onto physical embedded micro-UAV hardware platforms. These future deployments will explicitly incorporate unmodeled aerodynamic disturbances, such as continuous wind shear and localized atmospheric turbulence, while extending the core formulation to tackle dynamic multi-target trajectory tracking and complex autonomous multi-goal navigation tasks. Additionally, we will investigate the integration of adaptive goal-conditioned exploration strategies and hybrid model-based/model-free DRL architectures to further optimize real-time applicability and structural robustness in practical autonomous robotic systems.

References

[1] Milics, G.: Application of uavs in precision agriculture. In: International Climate Protection, pp. 93–97. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-03816-8_13

[2] Ekechi, C.C., Elfouly, T., Alouani, A., Khattab, T.: A survey on uav control with multi-agent reinforcement learning. Drones **9**(7), 484 (2025) https://doi.org/10.3390/drones9070484

[3] Gugan, G., Haque, A.: Path planning for autonomous drones: Challenges and future directions. Drones **7**(3), 169 (2023) https://doi.org/10.3390/drones7030169

[4] Sadi, M.A., Jamali, A., Abang Kamaruddin, A.M.N.: Optimizing uav performance in turbulent environments using cascaded model predictive control algorithm and pixhawk hardware. Journal of the Brazilian Society of Mechanical Sciences and Engineering **47**(8), 396 (2025) https://doi.org/10.1007/s40430-025-05693-9

[5] Farzaneh, S., Salimi Sartakhti, J.: Designing a model for data stream classification using reinforcement learning and stochastic gradient descent. Soft Computing Journal **12**(2), 2–15 (2024) https://doi.org/10.22052/scj.2023.248781.1124

[6] Raouf Moghadam, M., Ebrahimi, M.: An overview of deep learning applications in the aerospace industry. Soft Computing Journal **11**(2), 104–117 (2023) https://doi.org/10.22052/scj.2023.243422.1036

[7] Tong, G., Jiang, N., Biyue, L., Xi, Z., Ya, W., Wenbo, D.: Uav navigation in high dynamic environments: A deep reinforcement learning approach. Chinese Journal of Aeronautics **34**(2), 479–489 (2021) https://doi.org/10.1016/j.cja.2020.05.011

[8] Bai, Y., Zhao, H., Zhang, X., Chang, Z., Jäntti, R., Yang, K.: Toward autonomous multi-uav wireless network: A survey of reinforcement learning-based approaches. IEEE Communications Surveys & Tutorials **25**(4), 3038–3067 (2023) https://doi.org/10.1109/COMST.2023.3323344

[9] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International Conference on Machine Learning, pp. 1861–1870 (2018). https://doi.org/10.48550/arXiv.1801.01290 . Pmlr

[10] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017) https://doi.org/10.48550/arXiv.1707.06347

[11] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2015) https://doi.org/10.48550/arXiv.1509.02971

[12] Pan, L., Cai, Q., Huang, L.: Softmax deep double deterministic policy gradients. Advances in neural information processing systems **33**, 11767–11777 (2020) https://doi.org/10.48550/arXiv.2010.09177

[13] Futuhi, E., Karimi, S., Gao, C., Müller, M.: Etgl-ddpg: a deep deterministic policy gradient algorithm for sparse reward continuous control. arXiv preprint arXiv:2410.05225 (2024) https://doi.org/10.48550/arXiv.2410.05225

[14] Bouabdallah, S., Murrieri, P., Siegwart, R.: Design and control of an indoor micro quadrotor. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA) (2004). https://doi.org/10.1109/ROBOT.2004.1302409 . IEEE. https://www.research-collection.ethz.ch/bitstreams/842c51a1-6292-4a0d-a29d-4e60b41ce5ab/download

[15] Bouabdallah, S.: Design and control of quadrotors with application to autonomous flying. PhD thesis, EPFL (2007). https://infoscience.epfl.ch/entities/publication/4fbc8fc8-1a21-4a08-a24d-98f8302630f8

[16] Maaruf, M., Abubakar, A.N.u., Gulzar, M.M.: Adaptive backstepping and sliding mode control of a quadrotor. Journal of the Brazilian Society of Mechanical Sciences and Engineering (2024) https://doi.org/10.1007/s40430-024-05188-z

[17] Sarafrazi, P., Taher, S.A., Akhavan, A.: Suboptimal backstepping controller design for controlling shunt active power filter to compensate harmonics using the whale optimization algorithm. Soft Computing Journal **14**(2), 20–37 (2026) https://doi.org/10.22052/scj.2024.255064.1252

[18] Rubí, B., Morcego, B., Pérez, R.: Deep reinforcement learning for quadrotor path following with adaptive velocity. Autonomous Robots (2020) https://doi.org/10.1007/s10514-020-09951-8

[19] Rubí, B., Morcego, B., Pérez, R.: Quadrotor path following and reactive obstacle avoidance with deep reinforcement learning. Journal of Intelligent & Robotic Systems **103**, 62 (2021) https://doi.org/10.1007/s10846-021-01491-2

[20] Hu, W., Yang, Y., Liu, Z.: Deep deterministic policy gradient (ddpg) agent-based sliding mode control for quadrotor attitudes. Drones **8**(3), 95 (2024) https://doi.org/10.3390/drones8030095

[21] Caffyn Yuste, P., Iglesias Martínez, J.A., Miguel, M.A.: Simulation-based evaluation of model-free reinforcement learning algorithms for quadcopter attitude control and trajectory tracking. Neurocomputing **608**, 128362 (2024) https://doi.org/10.1016/j.neucom.2024.128362

[22] Fujimoto, S., Hoof, H., Meger, D.: Addressing function approximation error in actor-critic methods. In: International Conference on Machine Learning (ICML) (2018). https://doi.org/10.48550/arXiv.1802.09477 . https://arxiv.org/abs/1802.09477

[23] Guo, S., Zhang, X., Zheng, Y., Du, Y.: An autonomous path planning model for unmanned ships based on deep reinforcement learning. Sensors **20**(2), 426 (2020) https://doi.org/10.3390/s20020426

[24] Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P.: Trust region policy optimization. In: International Conference on Machine Learning, pp. 1889–1897 (2015). https://doi.org/10.48550/arXiv.1502.05477 . PMLR

[25] Tan, L., Sun, L., Cao, B., Xia, J., Xu, H.: Research on weighted energy consumption and delay optimization algorithm based on dual-queue model. IET Communications **18**(1), 81–95 (2024) https://doi.org/10.1049/cmu2.12710

[26] Wang, L., Wang, Y.-X., Li, J.-K., Liu, Y., Pi, J.-T.: Adaptive traffic signal control method based on offline reinforcement learning. Applied Sciences **14**(22), 10165 (2024) https://doi.org/10.3390/app142210165