Analysis of Reinforcement Learning-Based Altitude Control for a UAV Landing on a Moving Target under High Disturbance

Jad Alsaayed[1] and Hassan Noura[2]

***Abstract*— This paper presents a reinforcement learning (RL)–based altitude controller for a coaxial octorotor UAV performing landing on a moving platform under sudden, high-magnitude disturbance. The Deep Deterministic Policy Gradient (DDPG) algorithm is employed to learn vertical thrust commands directly from interaction with the environment, while proportional-integral-derivative (PID) loops handle horizontal positioning. A custom reward penalizes both residual altitude error—including steady-state error—and excessive thrust changes. In MATLAB/SIMULINK simulations, the RL controller maintains precise descent trajectories, drives steady-state error to near zero, and adapts to both severe and mild gusts without retuning, outperforming a conventional PID in all tested scenarios. These results demonstrate the promise of purely RL-based altitude control for robust UAV operations in challenging wind conditions.**

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) are increasingly used for applications ranging from package delivery to infrastructure inspection and search-and-rescue, where precision landings on moving platforms under turbulent conditions are essential. Traditional PID controllers can maintain altitude in light winds but often exhibit steady-state error, overshoot or oscillations when faced with strong or sudden disturbance. Model-free reinforcement learning (RL) offers a data-driven alternative: by learning control policies directly from trial-and-error interactions with the environment, a RL agent can adapt thrust commands to complex, time-varying disturbance without an explicit aerodynamic model.

Early RL controllers demonstrated the ability to handle nonlinearities and uncertainties. One Fuzzy Reinforcement Learning Control (FRLC) scheme uses fuzzified sliding surfaces and local velocities as inputs to an RL agent that outputs smooth actuator voltages. FRLC learns to map these inputs to control signals, achieving precise trajectory tracking without manually designed rules [1].

In aerial robotics, end-to-end proximal policy optimization (PPO) policies have been shown to directly command low-level thrust and torque for all six degrees of freedom, learning in simulation under unmodeled disturbances and matching the performance of manually-tuned proportional-derivative (PD) loops across hover and descent tasks by penalizing altitude tracking error and control effort in the reward [2]. Alternatively, a hierarchical scheme integrates a conventional PID lateral controller with a DDPG agent that adaptively tunes the PID gains via corrective-feedback heuristics at each time step, yielding smoother, more reliable landings on moving decks—even with model-free training—though earlier demonstrations focused on mild, periodic disturbances [3]. Other approaches have treated altitude control outside the learned policy (for instance, omitting the Z-axis from the action space entirely to simplify safety), which precludes fully integrated three-dimensional learning [4].

While prior work shows RL handling mild or periodic disturbances, none tackle abrupt, high-magnitude gusts while simultaneously minimizing control effort, trajectory tracking error, and steady-state error. To address this, a DDPG–based altitude controller for a coaxial octorotor is presented, trained in simulation to reject sudden gusts while maintaining precise altitude tracking, minimal steady-state error, and low control effort, with horizontal motion handled by classical PID loops.

The paper is organized as follows. Section II introduces the coaxial octorotor's nonlinear dynamics. Section III presents the DDPG-based altitude-control framework. Section IV presents simulation results on hyperparameter tuning, disturbance rejection, and PID comparison. Finally, Section V concludes and outlines future work.

II. NONLINEAR MODEL OF A COAXIAL OCTOROTOR

The UAV dynamics are modeled using the Euler-Lagrange formalism [5] to capture its behavior in roll (ϕ), pitch (θ), and yaw (ψ) (see Fig. 1). The UAV parameters are: mass $m = 1.6$ kg, arm length $l = 0.23$ m, thrust coefficient $K_f = 3.5 \times 10^{-5}$ N·s^2/rad^2, moment coefficient $K_t = 3 \times 10^{-7}$ N·m/rad^2, with moments of inertia $I_{xx} = I_{yy} = 0.0255$ kg·m^2 and $I_{zz} = 0.0388$ kg·m^2.

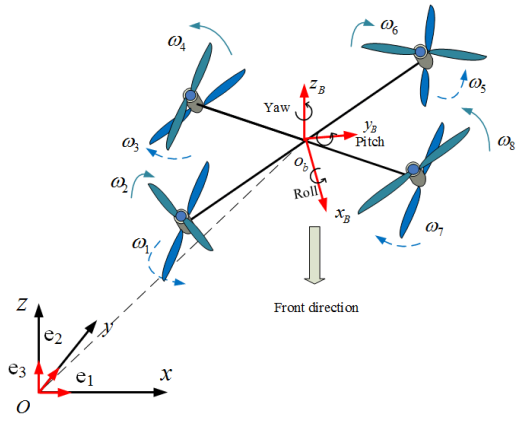

Fig. 1. Rotational axes and angles of the coaxial octorotor UAV.

[1]Jad Alsaayed is with the Faculty of Engineering, Lebanese University, Beirut, Lebanon jadhsaayed@gmail.com

[2]Hassan Noura is with Aix-Marseille University, CNRS, LIS (UMR 7020), Avenue Escadrille Normandie-Niemen, F-13397 Marseille Cedex 20, France hassan.noura@univ-amu.fr

The translational dynamics in the Earth Frame (EF) are:

$$\ddot{x} = \frac{(\cos\phi \, \sin\theta \, \cos\psi + \sin\phi \, \sin\psi)}{m} U_z, \tag{1}$$

$$\ddot{y} = \frac{(\cos\phi \, \sin\theta \, \sin\psi - \sin\phi \, \cos\psi)}{m} U_z, \tag{2}$$

$$\ddot{z} = \frac{(\cos\phi \, \cos\theta)}{m} U_z - g, \tag{3}$$

where g is gravitational acceleration and U_z is the total thrust force.

The rotational dynamics about the body axes are given by:

$$\ddot{\phi} = \frac{I_{yy} - I_{zz}}{I_{xx}} \dot{\theta}\dot{\psi} - \frac{J_r}{I_{xx}} \dot{\theta}\Omega + \frac{1}{I_{xx}} \tau_\phi, \tag{4}$$

$$\ddot{\theta} = \frac{I_{zz} - I_{xx}}{I_{yy}} \dot{\phi}\dot{\psi} + \frac{J_r}{I_{yy}} \dot{\phi}\Omega + \frac{1}{I_{yy}} \tau_\theta, \tag{5}$$

$$\ddot{\psi} = \frac{I_{xx} - I_{yy}}{I_{zz}} \dot{\phi}\dot{\theta} + \frac{1}{I_{zz}} \tau_\psi, \tag{6}$$

which assume small attitude angles so that $\dot{\phi}$, $\dot{\theta}$, and $\dot{\psi}$ approximate the body-fixed angular velocities. This is consistent with the use of PID controllers for stabilization.

The total thrust and torques are computed as:

$$U_z = F_{12} + F_{34} + F_{56} + F_{78}, \tag{7}$$

$$\tau_\phi = \frac{(F_{78} + F_{56} - F_{12} - F_{34})\, l\sqrt{2}}{2}, \tag{8}$$

$$\tau_\theta = \frac{(F_{34} + F_{56} - F_{12} - F_{78})\, l\sqrt{2}}{2}, \tag{9}$$

$$\tau_\psi = (\tau_2 + \tau_3 + \tau_6 + \tau_7) - (\tau_1 + \tau_4 + \tau_5 + \tau_8). \tag{10}$$

Each rotor i produces:

$$F_i = K_f \, \omega_i^2, \quad \tau_i = K_t \, \omega_i^2, \tag{11}$$

and for a coaxial pair:

$$F_{ij} = \sigma(F_i + F_j), \tag{12}$$

with σ accounting for coaxial effects.

The control strategy uses PID controllers for horizontal positioning and attitude stabilization, while an RL controller manages altitude control.

III. DRL and DDPG for UAV Control

Deep Reinforcement Learning (DRL) leverages deep neural networks to address complex, high-dimensional tasks [6]. In this work, the Deep Deterministic Policy Gradient (DDPG) algorithm—a model-free, off-policy method—is used to control the altitude of a coaxial octorotor UAV. DDPG is particularly well suited for continuous action spaces, making it a powerful choice for generating smooth thrust commands in real time. It employs an actor-critic framework where the actor maps states to continuous actions and the critic estimates the action-value function $Q(s, a)$ [7].

A. MDP Formulation

The UAV control problem is modeled as a Markov Decision Process with:

- **State Space** S: UAV conditions, including measured altitude Z_{measured}, desired altitude Z_{desired}, measured vertical velocity V_{measured}, and desired vertical velocity V_{desired}.
- **Action Space** A: Continuous control input U_z for altitude adjustments.
- **Transition Dynamics** T: The UAV's dynamic response.
- **Reward Function** R: Designed to minimize altitude tracking error and penalize excessive control effort.

The objective is to learn a policy that maximizes the cumulative reward $R = \sum_{t=0}^{\infty} \gamma^t r_{t+1}$ [6] where γ is the discount factor.

B. Network Architecture and Exploration Strategy

Both the actor and critic networks consist of two fully connected layers with 256 neurons each and ReLU activations. The critic takes the 4D state and action as input to estimate $Q(s, a)$, while the actor maps the state to U_z and scales it via a tanh activation. Exploration is facilitated by adding Ornstein-Uhlenbeck noise to the actor's output, with the noise magnitude decaying over time.

C. DDPG Algorithm Overview

DDPG uses experience replay [8] and target networks to stabilize training. The critic minimizes the Temporal Difference (TD) error:

$$L = \mathbb{E}\left[(y_i - Q(s_i, a_i \mid \theta^Q))^2\right], \tag{13}$$

with targets computed as:

$$y_i = r_{i+1} + \gamma Q'\big(s_{i+1}, \pi(s_{i+1} \mid \theta^{\pi'}) \mid \theta^{Q'}\big). \tag{14}$$

Target networks Q' and π' are updated using a soft update rule:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \quad \theta^{\pi'} \leftarrow \tau\theta^\pi + (1-\tau)\theta^{\pi'}, \tag{15}$$

with a small τ (e.g., 0.001). The ADAM optimizer [9] is used for parameter updates, and the actor is updated using deterministic policy gradients [10].

D. Reward Function Design

The reward function evaluates the UAV's performance at each time step, guiding the agent to achieve accurate trajectory tracking. Two reward function formulations were explored to enhance control stability and precision:

1) **Initial Reward Function:** The altitude tracking error is minimized based on the difference between the measured altitude z_m and the desired altitude z_d:

$$r_1(z_m, z_d) = -|z_m - z_d| \tag{16}$$

2) **Control Effort and Integral Error Penalties:** Penalties for excessive control effort U_z and the integral of the tracking error I_e are added:

$$r_2(z_m, z_d, I_e, U_z) = -|z_m - z_d| - a \cdot |U_z| - b \cdot |I_e| \tag{17}$$

The integral of error I_e accumulates the altitude tracking error over time, penalizing prolonged deviations from the desired altitude. By addressing accumulated error, this term helps reduce steady-state errors and promotes smoother, more stable control.

In both reward formulations, a positive constant reward $+c$ is applied when the error $|z_m - z_d|$ falls below 0.01 meters to encourage precise altitude control:

$$\text{if} \quad |z_m - z_d| < 0.01, \quad r = r + c \tag{18}$$

where c is the positive reward constant, and a and b are tuning weights for penalizing control effort and integral error in the second reward function.

IV. Results and Discussion

A. Simple Altitude Control: Network and Reward Tuning

The RL agent was initially tested on a simple descent task, where the UAV controlled its altitude from 5 meters to ground level over 20 seconds, then hovered for 15 seconds. This test assessed the agent's basic descent and stabilization abilities, with the reference trajectory shown in Fig. 2.

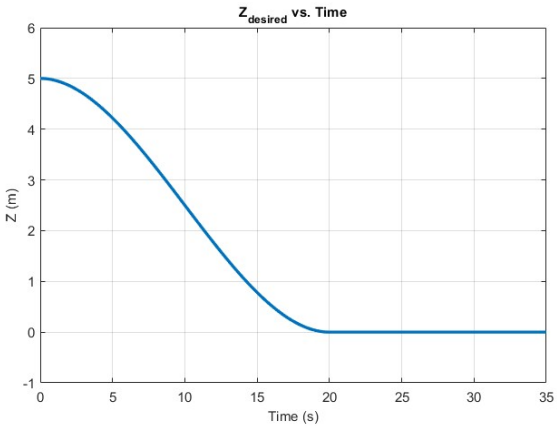

Fig. 2. UAV altitude reference.

1) Hyperparameter Configuration and Initial Testing: Preliminary hyperparameters were chosen based on prior studies and early experimental results. Initially, a smaller network (64 neurons) led to significant transient errors (Fig. 3), indicating premature convergence due to limited capacity. A larger configuration (400 neurons) demonstrated reduced tracking error but introduced minor hovering oscillations (Fig. 4).

The hyperparameters used in both the initial testing phase and the optimized configuration after tuning are presented in Table I.

2) Refining the Reward Function: The reward function was refined (as presented in (17)) to address steady-state error and oscillations. The resulting trajectory tracking improved significantly, as shown in Fig. 5, with Fig. 6 depicting the corresponding control input.

TABLE I

Initial and Optimized Hyperparameters for DDPG Training

Hyperparameter	**Symbol**	**Initial Value**	**Optimized Value**
Learning Rate (Actor)	α_{actor}	0.00004	0.0001
Learning Rate (Critic)	α_{critic}	0.0004	0.001
Noise Std. Dev.	σ_{noise}	0.8	0.4
Noise Decay Rate	σ_{decay}	0.0001	0.00025
Batch Size	N_{batch}	256	64
Replay Buffer Size	$\lvert\mathcal{B}\rvert$	10^6	10^6
Target Smooth Factor	τ	0.001	0.001
Step Time	Δt	0.1 s	0.1 s
Discount Factor	γ	0.97	0.92

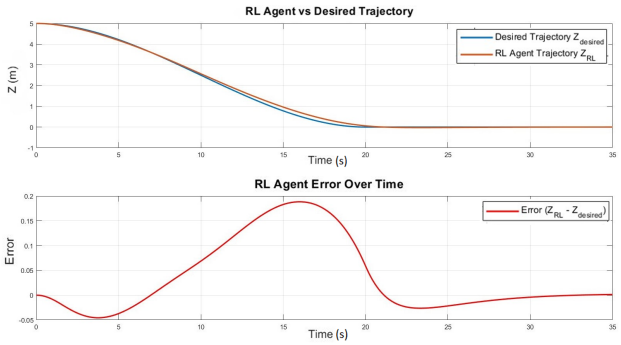

Fig. 3. Tracking error with 64 neurons per layer.

Fig. 4. Tracking error with 400 neurons per layer.

Fig. 5. Reduced steady-state error and oscillation with revised reward function.

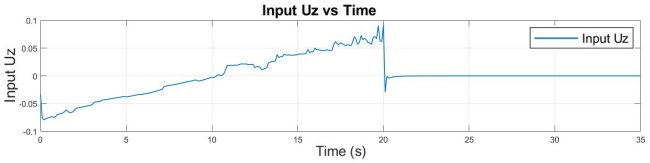

Fig. 6. Control input U_z with revised reward function.

3) Performance Evaluation and Robustness Testing: The agent was trained over 820 episodes, achieving a final reward of 10,371. Reward progression and Q-value trends are shown in Fig. 7, where some instability in the training process is observed. This instability, primarily caused by the high number of neurons in the neural network (400 neurons per layer), resulted in overfitting and an extended training duration. In subsequent training sessions, various network configurations were tested, leading to improved training stability without sacrificing model capacity.

In addition, the RL controller faced significant challenges under a disturbance injection from 5 to 20 seconds, as shown in Figs. 8 and 9. During this period, the controller initially lost stability, highlighting the need for further refinements to the control law. These refinements were focused on enhancing the system's robustness to high disturbance, ensuring that the controller could handle more dynamic and unpredictable environmental conditions while maintaining stability.

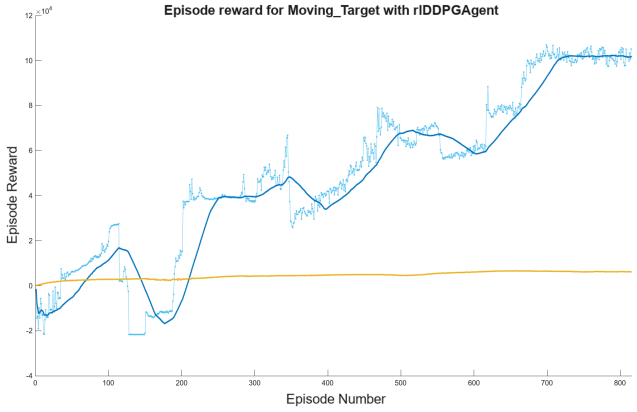

Fig. 7. Training performance showing episode rewards (light blue), moving-average rewards (dark blue), and critic Q-value estimates (orange) over episodes.

Fig. 8. Tracking error with Disturbance Injected between 5 and 20 seconds.

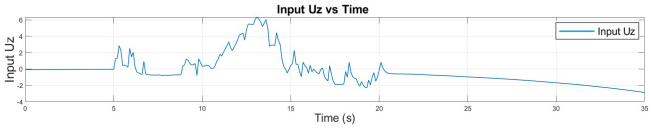

Fig. 9. Control input U_z of the RL Agent with Disturbance Injection.

B. Challenging Task Evaluation with Disturbance

To develop a robust control law for a coaxial octorotor UAV, the DDPG algorithm was refined through systematic hyperparameter tuning to optimize training efficiency and disturbance handling. Key adjustments included increased actor-critic learning rates, reduced discount factor, reduced noise levels, and smaller batch sizes, as summarized in Table I, which displays both initial and optimized hyperparameters. Additionally, a streamlined network architecture (as detailed in Section III-B) was implemented to balance complexity with faster convergence.

To further stabilize learning and accelerate convergence, reward clipping was applied to limit extreme negative values (Fig. 7), and the reward output was scaled by 0.01. This adjustment mitigated fluctuations in rewards and improved training stability:

$$r_2 = \max(r_2, -r_{\min}) \tag{19}$$

The UAV was evaluated on a complex 45-second reference trajectory in the Z plane, involving descents, hovering, and ascents, as shown in Fig. 10. To simulate real-world conditions, a wind disturbance model was applied during the task. The model consisted of a sinusoidal wind gust with a velocity amplitude of 3 m/s, along with random turbulence components varying between -5 m/s and $+5$ m/s. These wind velocity inputs were converted into equivalent disturbance accelerations applied to the UAV dynamics, which are shown in Fig. 11. The resulting disturbance accelerations were applied between 6 and 30 seconds, introducing significant challenges that required the RL controller to dynamically adjust and maintain altitude control. The total disturbance acceleration is modeled as:

$$a_z = \text{sinusoidal gust} + \text{sudden sharp change} \tag{20}$$

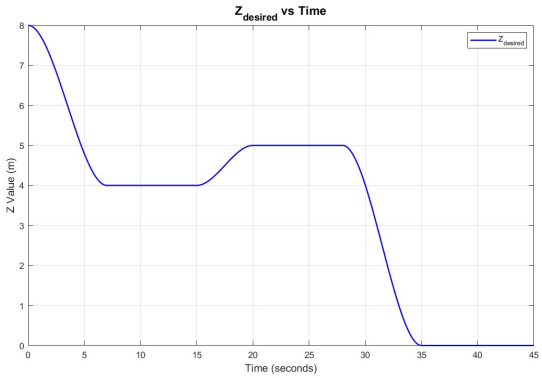

Fig. 10. UAV altitude reference.

Despite this disturbance, the RL controller maintained low tracking error, as demonstrated in Fig. 12. The corresponding control input U_z is shown in Fig. 13, highlighting the agent's adaptive response to the external disturbance. The agent was trained over 1,000 episodes, completing 450,000 steps in 25 hours, with Fig. 14 showing an upward trend in rewards, indicating that the agent successfully converged to an optimal policy. These results demonstrate that the tuning of hyperparameters, network configuration, and reward adjustments contributed significantly to the agent's robust

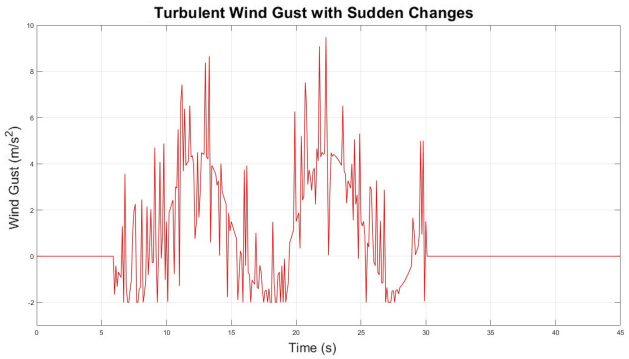

Fig. 11. Acceleration Disturbance Model Used During Evaluation

performance under high-disturbance conditions, reaching an optimal reward of 479.95.

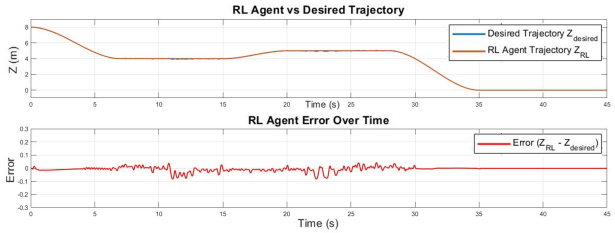

Fig. 12. RL agent trajectory and tracking error with disturbance.

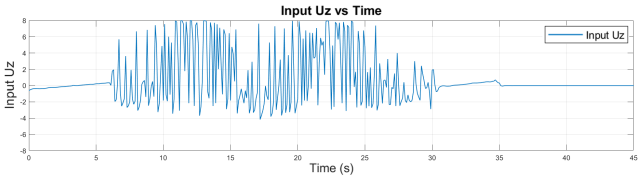

Fig. 13. Control input U_z of the RL Agent with Disturbance.

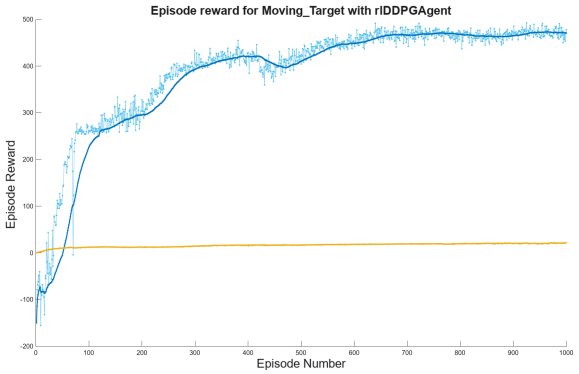

Fig. 14. Training performance with episode rewards and Q-value progression.

1) Testing on the Same Trajectory Without Disturbance: The control law was further evaluated on the same trajectory but without disturbance. The results, shown in Fig. 15, indicate that the RL controller maintained strong performance in the absence of disturbance, effectively tracking the desired trajectory. The corresponding control input U_z is illustrated in Fig. 16, demonstrating smooth and consistent adjustments throughout the task.

Fig. 15. RL agent trajectory and tracking error without Disturbance.

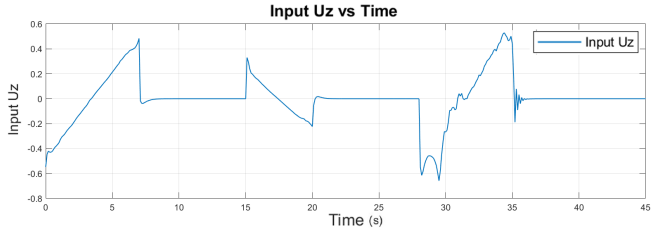

Fig. 16. Input U_z of the RL Agent without Disturbance.

C. Control Law for the Simple Task with Disturbance

The control law was retested on the same simple trajectory shown in Fig. 2, where the initial control approach had failed to compensate for the disturbance. In this scenario, a disturbance was introduced between 5 and 20 seconds. The RL agent successfully compensated for the disturbance and maintained accurate tracking throughout the task, as shown in Fig. 17. This demonstrates the robustness of the refined control law under conditions that previously caused instability. The corresponding control input U_z is shown in Fig. 18, highlighting the agent's adaptive response to the external disturbance.

Fig. 17. RL Agent trajectory and tracking error with Disturbance injected between 5 and 20 seconds.

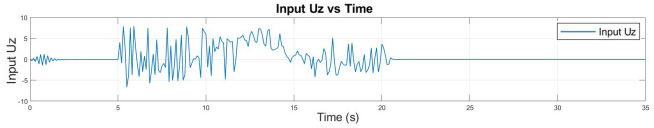

Fig. 18. Control input U_z of the RL Agent with Disturbance injected between 5 and 20 seconds.

D. Comparison with PID Controller for Altitude Control

A well-tuned PID controller was selected as the baseline since PID loops remain the industry standard for UAV altitude control. The RL-based controller's altitude tracking performance was compared to this PID baseline using the root-mean-square error (RMSE) of the altitude deviation. Figures 19 and 20 illustrate the PID controller's trajectory, tracking error, and thrust commands U_z under the same landing task (Figure 10) and disturbance profile (Figure 11). As reported in Table II, the RL policy consistently achieves lower RMSE than the PID controller across all scenarios. This improvement stems from training directly in an environment with strong, mixed-frequency gusts, making the RL agent robust to both the training disturbance profile and to similar stochastic or periodic wind conditions in practice.

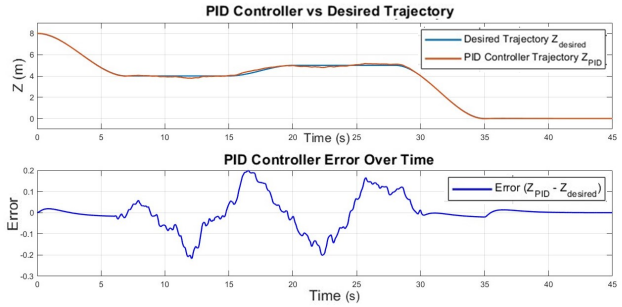

Fig. 19. PID controller trajectory and tracking error with disturbance.

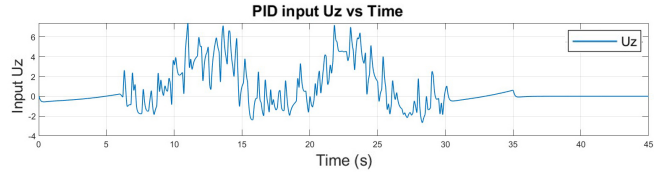

Fig. 20. Control input U_z of the PID controller with Disturbance.

TABLE II

TASK RMSE COMPARISON BETWEEN RL AND PID

Task Description	RL RMSE	PID RMSE
Simple Landing, No Disturbance	0.0027	0.0041
Simple Landing, With Disturbance	0.0511	0.0635
Complex Landing, No Disturbance	0.0042	0.006
Complex Landing, With Disturbance	0.0391	0.0813

E. Practical Implications

Since the RL controller was trained on a high-disturbance profile, its tracking performance may degrade under gusts stronger than those encountered during training. However, within the tested disturbance range, it delivers reliable altitude tracking in both severe and calm conditions. Under light or no disturbance, the controller generates smooth thrust commands, improving energy efficiency and reducing mechanical wear—extending flight time and lowering maintenance costs.

Notably, a single parameter set, tuned for the selected disturbance profile, ensures consistent performance across all tested scenarios, eliminating the need for PID retuning.

V. CONCLUSION

This paper presents a DDPG-based RL controller for UAV landing altitude control on a coaxial octorotor, demonstrating accurate descent trajectories and energy-efficient thrust commands that minimize steady-state error under both extreme gusts and calm conditions. The custom reward function—penalizing residual altitude error and large thrust changes—enables smooth, disturbance-adapted landings without manual retuning. A key limitation is the sensitivity of DDPG, which required extensive hyperparameter tuning, reward function adjustments, and prolonged training. Future work will extend the RL approach to full six-degree-of-freedom control and develop methods to manage the increased state–action complexity and ensure safe exploration.

REFERENCES

[1] M. Q. Zaman and H.-M. Wu, "Fuzzy reinforcement learning based trajectory-tracking control of an autonomous mobile robot," in *2022 22nd International Conference on Control, Automation and Systems (ICCAS)*, 2022, pp. 840–845.

[2] Z. Jiang and A. F. Lynch, "Quadrotor motion control using deep reinforcement learning," *Journal of Unmanned Vehicle Systems*, vol. 9, no. 4, pp. 234–251, 2021. [Online]. Available: https://doi.org/10.1139/juvs-2021-0010

[3] L. Wu, C. Wang, P. Zhang, and C. Wei, "Deep reinforcement learning with corrective feedback for autonomous uav landing on a mobile platform," *Drones*, vol. 6, p. 238, 09 2022.

[4] J. Xie, X. Peng, H. Wang, W. Niu, and X. Zheng, "Uav autonomous tracking and landing based on deep reinforcement learning strategy," *Sensors*, vol. 20, no. 19, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/19/5630

[5] M. Saied, B. Lussier, I. Fantoni, H. Shraim, and F. Clovis, "Active versus passive fault-tolerant control of a redundant multirotor uav," *Aeronautical Journal -New Series-*, vol. 124, 11 2019.

[6] R. Sutton and A. Barto, "Reinforcement learning: An introduction," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 1054–1054, 1998.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2019. [Online]. Available: https://arxiv.org/abs/1509.02971

[8] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Mach. Learn.*, vol. 8, no. 3–4, p. 293–321, May 1992. [Online]. Available: https://doi.org/10.1007/BF00992699

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. [Online]. Available: https://arxiv.org/abs/1412.6980

[10] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 1. Bejing, China: PMLR, 22–24 Jun 2014, pp. 387–395. [Online]. Available: https://proceedings.mlr.press/v32/silver14.html

[11] D. Weber, M. Schenke, and O. Wallscheid, "Steady-state error compensation for reinforcement learning-based control of power electronic systems," *IEEE Access*, vol. 11, pp. 76 524–76 536, 2023.