Advances in Consumer Research
https://acr-journal.com/
Volume-2 | Issue-5 | Nov 2025

Original Researcher Article

Real-Time Performance Optimization of Uam Propulsion Sys-tem Using Ddpg Algorithm

Kim DongJun[1], Heo DoHyun[1] and Jie MinSeok[2*]
[1]Department of Aeronautical Systems Engineering, Hanseo University, Republic of Korea
[2]Department of Avionics Engineering, Hanseo University, Republic of Korea

Received:
10/09/2025
Revised:
25/10/2025
Accepted:
17/11/2025
Published:
23/11/2025

ABSTRACT
Com Propulsion system for Urban Air Mobility (UAM) vehicles and characterized by non-linear system dynamics, have traditionally relied on classical PID controllers for con-trol Optimal tuning of PID gains for these nonlinear systems is commonly derived from empirical process, or for optimal control, commonly derived using methods such as the Riccati equation or Linear Quadratic Regulator (LQR), which linearize the sys-tem around an operating point. These approaches often lose optimality when the UAM propulsion system deviates significantly from the linearization point, which is fre-quent during various flight phases. Physical characteristics of the engine change over time due to aging and wear, necessitating manual retuning and incurring additional maintenance costs to sustain optimal performance. To address these critical challenges and maintain optimal control performance continuously for UAM propulsion, this paper proposes a reinforcement learning-based approach. Specifically, the Deep De-terministic Policy Gradient (DDPG) algorithm is applied to implement an adaptively optimized PID controller, enabling real-time performance optimization of the propul-sion system. The proposed method is validated through comprehensive simulations conducted in Matlab Simulink, demonstrating its effectiveness in maintaining optimal and adaptive control under the highly nonlinear and time-varying operational condi-tions characteristic of UAM propulsion systems.

Keywords: Reinforcement learning; Deep deterministic policy gradient; UAM; Propul-sion system; Artificial intelligence.

INTRODUCTION:

Propulsion Systems, characterized by inherently nonlinear system dynamics, have traditionally been controlled using classical methods such as Proportional-Integral-Derivative (PID) controllers. Owing to their simplicity and effectiveness, PID controllers have been extensively applied across a wide range of systems for decades. Optimal tuning of PID gains for such nonlinear systems is typically achieved through methods like the Riccati equation or the Linear Quadratic Regulator (LQR), which focus on deriving optimal parameters by linearizing the nonlinear system around a specific operating point. However, such linearization-based approaches inherently assume that the system remains near the linearization point, and thus, their optimality and performance guarantees deteriorate once the system deviates significantly from this region

The physical characteristics of the UAM propulsion system evolve over time due to factors such as engine aging and wear. This phenomenon often necessitates manual retuning of the controller to maintain acceptable performance, which introduces additional operational costs, increases downtime, and complicates the overall maintenance process. These challenges underscore the inherent limitations of conventional control schemes in effectively adapting to the highly nonlinear and time-varying behaviors critical for safe and efficient UAM operations. The intricate dynamics of propulsion systems, coupled with the stringent performance requirements of UAM, emphasize the pressing need for more efficient, autonomous, and adaptive control methodologies.

In response to these critical challenges advanced control techniques leveraging artificial intelligence have been investigated. Examples include fuzzy control by Jung-Byung-In [1] and neural network-based control by Kim-Dae-Ki [2]. These methods demonstrated superior performance in modelling and controlling engine nonlinearities and uncertainties compared to conventional linear control approaches. Building upon this lineage of advanced control techniques, this paper applies a reinforcement learning-based control strategy to optimize the PID parameters of a propulsion system.
Prior to the application of artificial neural networks, traditional reinforcement learning methodologies relied on table-based approaches, such as Q-Tables, to store all possible state-action combinations. While effective in simple environments, this method suffered from the "curse of dimensionality" in complex environments where state and action spaces are high-dimensional and require continuous value outputs, demanding immense storage and computational resources. This limitation made practical application to real-world systems nearly impossible and restricted their generalization ability to unseen states, requiring direct experience of every possible scenario for learning to occur. Following the

introduction of DeepMind's Deep Q-Network (DQN) [3] in 2013, reinforcement learning achieved a significant milestone in solving complex decision-making problems by using deep neural networks to approximate Q-functions. This enabled efficient learning in large-scale state spaces and demonstrated high generalization capabilities.

The introduction of DeepMind's Deep Q-Network (DQN) in 2013, RL has marked a significant milestone in solving complex decision-making problems. DQN is inherently limited to discrete action space, making it less suitable for precise control of continuous systems like jet engines. To address this limitation for continuous control problems, research builds upon the Deep Deterministic Policy Gradient (DDPG) algorithm, which is designed for continuous action spaces and directly learns deterministic policies. DDPG's actor-critic architecture facilitates stable neural network training and robust learning in nonlinear dynamic systems where continuous control outputs are essential. This deep reinforcement learning approach enables the controller to learn optimal control policies directly from interaction with the environment, adapting dynamically to changes in Propulsion System characteristics and operating conditions, thereby leading to real-time performance optimization.

In 2021, Junru Yang and Weifeng Peng [4] demonstrated the superior robustness of a Deep Deterministic Policy Gradient (DDPG) [5] based PID controller through their research on Cooperative Adaptive Cruise Control (CACC) systems using deep learning-based PID control techniques. Similarly, in 2023, Steven Bandong [6] proved that a DDPG-based reinforcement learning approach could learn optimal PID parameters for a Rubber Tired Gantry Crane, exhibiting competent performance not only across various reference trajectories but also under diverse system parameter conditions such as container mass and rope length.

This study simplifies and reconstructs the Actor neural network to align with the inherent properties of PID control, thereby limiting the range of actions. This reconfigured Actor network helps to maintain more stable and predictable policy updates, leading to improved learning stability and performance. Proposed control framework is validated through comprehensive simulations conducted within the Matlab Simulink environment, specifically tailored to model a UAM propulsion system. These simulations are designed to demonstrate the efficacy and robustness of the DDPG-based adaptive PID controller in maintaining optimal performance across various nonlinear and time-varying operational conditions. This approach is anticipated to effectively address the inherent nonlinearities of propulsion systems and sustain optimal control performance across a broad spectrum of UAM operating conditions, paving the way for more autonomous and reliable UAM propulsion systems.

Propulsion system

The dynamic control equation for a jet engine model is derived from the equilibrium equations, such as those for power, energy, and flow, applied to each component. The resulting coupled nonlinear equation is obtained from a component matching program solution relative to a reference operating point [7]. Consequently, the engine's state equation can be expressed as shown in Equation (1).

$$\dot{x} = f(x, u) \qquad (1)$$

In Equation (1), the state vector of the engine is denoted by x, and the control input vector is denoted by u. At any typical operating $\dot{x}$ = 0. By considering this condition and neglecting higher-order differential terms during a Taylor series expansion, a linear relationship, as presented in Equation (2), can be derived.

$$\Delta\dot{x} = \frac{\partial\dot{x}}{\partial x}|_0 \Delta x + \frac{\partial\dot{x}}{\partial x}|_0 \Delta u \qquad (2)$$

In Equation (2), $\Delta x = x - x_0$, $\Delta u = u - u_0$. Since the actual system includes several state variables and input variables, the form of vector matrices, such as Equation (3), is generally established near one operating point.

$$\Delta\dot{x} = A\Delta x + B\Delta u \qquad (3)$$

If there are n state variables and m control input variables, i is a subscript representing a row and j is a column, each matrix can be expressed as Equation (4).

$$\dot{x}_p(t) = A_p x_p(t) + B_p u_p(t) \qquad (4)$$
$$A_p = \left[\frac{\partial \dot{x}_i}{\partial x_j}\right]_0 \quad where \left(\frac{i=1,n}{j=1,n}\right), \; B_p = \left[\frac{\partial \dot{x}_i}{\partial x_j}\right]_0 \quad where \left(\frac{i=1,n}{j=1,n}\right)$$

In Equation (5), $x_p = [x_{p1}\, x_{p2}\, x_{p3}]^T$: state variable vector, and each variable is as follows.
x_{p1}: Compressor rotor rotation speed

x_{p2} : Turbine inlet temperature

x_{p3} : Compressor outlet pressure

In this paper, compressor rotation speed (Δx_1), turbine inlet temperature (Δx_2), compressor outlet pressure (Δx_3), and fuel flow rate (u) were used as state variable vectors.

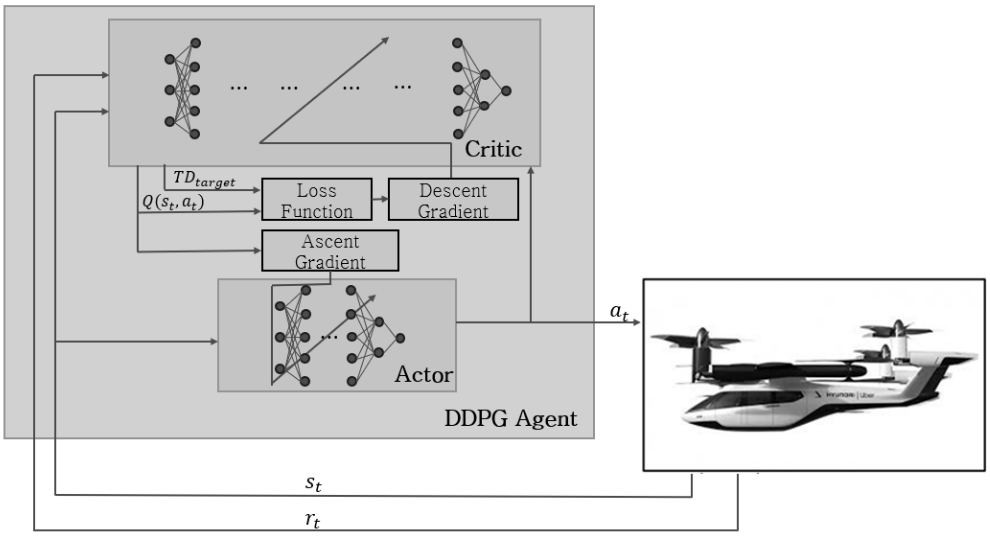
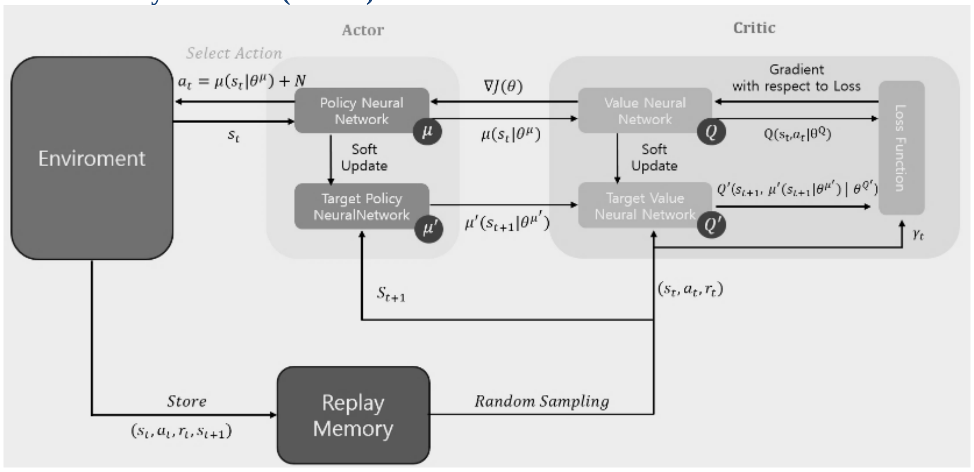


Figure 1. Structure of DDPG Algorithm

3. Deep Deterministic Policy Gradient (DDPG)



The deterministic nature of the DDPG Actor leads it to output a specific action, which is then evaluated by the Critic. This deterministic characteristic is advantageous for applications involving continuous action spaces. Figure 2 depicts the diagram of the DDPG algorithm.

The fundamental principle of the DDPG algorithm is to maximize the expected value of the Return defined in equation (5).

$$Return = G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} \cdots \tag{5}$$

$$Q(s,a) = E[G_t] = \int_{a_t:a_\infty} G_t P(s_{t+1}, a_{t+1}, s_{t+2}, \cdots | s_t, a_t) da_t : a_\infty \tag{6}$$

From equation (6), the Bellman equation (7) is derived by applying Bayesian theory and the Markov chain property.

$$Q(s,a) = E[G_t]$$
$$= \int_{s_{t+1}:a_{t+1}} (r_t + \gamma Q(s_{t+1}, a_{t+1})) P(s_{t+1}, a_{t+1} | s_t, a_t) ds_{t+1} : a_{t+1} \tag{7}$$

According to the law of large numbers, when functions following a certain probability distribution are randomly sampled and averaged, the result converges to the expected value of the function. This relationship is expressed in equation (8).

$$Q^*(s,a) \approx \bar{Q}_N = \frac{1}{N}\sum_{i=1}^{N} r_t^i + \gamma Q\left(s_{t+1}^i, a_{t+1}^i\right) \tag{8}$$

Equation (8) can be expressed as equations (9), (10), and (11).

$$Q^*(s,a) = \frac{1}{N}(\bar{Q}_{N-1}(N-1) + r_t^N + \gamma Q(s_{t+1}^N, a_{t+1}^N)) \tag{9}$$

$$Q^*(s,a) = \bar{Q}_{N-1} + \frac{1}{N}(r_t^N + \gamma Q(s_{t+1}^N, a_{t+1}^N) - \bar{Q}_{N-1}) \tag{10}$$

$$Q^*(s,a) = (\frac{N-1}{N})\bar{Q}_{N-1} + \frac{1}{N}(r_t^N + \gamma Q(s_{t+1}^N, a_{t+1}^N)) \tag{11}$$

TD_{Target} is computed using the Target Policy Neural Network (μ') and the Target Value Neural Network (Q'), as illustrated in Fig. 2, and is mathematically expressed by Equation (12).

$$TD_{Target} = r_t + \gamma Q'(s_{t+1}, \mu'(s_{t+1})) \tag{12}$$

r_t : Reward at [t]

s_{t+1} : State at [t+1]

Parameters(θ^Q)of the Value Neural Network(Q) are trained by minimizing the loss defined in Equation (13) through gradient descent using backpropagation.

$$LossQ = \frac{1}{N}\sum_{i=1}^{N}(TD_{Target} - Q(s_i, a_i|\theta^Q))^2 \tag{13}$$

N : Size of mini batch

s_i : State at [t]

The parameters(θ^μ)of the policy are trained via gradient ascent to maximize the corresponding Q-value. The policy converges to the optimal policy as described by Equation (15).

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_a Q(s_i, \mu(s_i)|\theta^Q)\nabla_{\theta^\mu}\mu(s_i|\theta^\mu) \tag{14}$$

$$\theta^\mu{}_{t+1} = \theta^\mu{}_t + \alpha\nabla_\theta J(\theta) \tag{15}$$

The parameters of the target networks $\theta^{Q'}$and $\theta^{\mu'}$are updated to follow the network parameters $\theta^Q$and θ^μthrough a soft update method, as expressed in Equation (16).

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} \tag{16}$$

τ :

Soft update factor

4. Simulation

Authors should discuss the results and how they can be interpreted from the perspective of previous studies and of the working hypotheses. The findings and their implications should be discussed in the broadest context possible. Future research directions may also be highlighted.

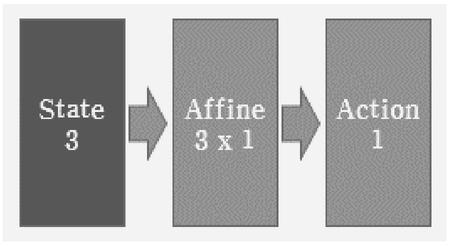


Figure 3. Actor network

The Actor network receives a State input consisting of three features. These features represent critical information for decision-making, such as the error between the reference and the current output, the integral of this error, and its derivative. Figure 4 illustrates the detailed structure of the states employed within the simulation environment.

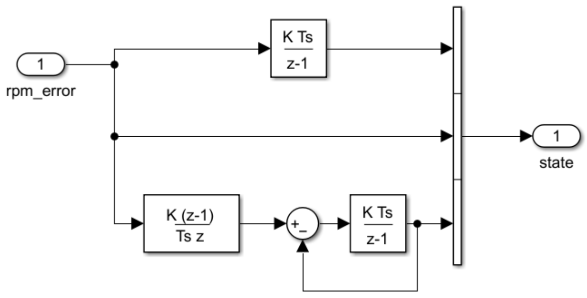


Figure 4. State production block

Internally, the network employs three distinct weight parameters, each corresponding to one of the input state features. The output of the Actor is then derived from the summation of the product of each input state feature with its respective weight. This can be formally expressed as:

$$\text{Action} = w_P \cdot S_P + w_I \cdot S_I + w_D \cdot S_D \quad (17)$$

where Sp,Si,Sd are the three state inputs and wp,wi,wd are the corresponding learned weights. This architectural choice directly mirrors the operation of a PID controller, where proportional, integral, and derivative gains Kp, Ki, Kd are multiplied by their respective error terms and summed to produce a control output. During the reinforcement learning process, the three internal parameters of the Actor network are observed to learn characteristics akin to those of Kp, Ki, and Kd gains, respectively. This behavior allows the Actor to implicitly capture the balancing act of a PID controller, providing immediate, accumulated, and predictive control responses.

The Critic network, as depicted in Figure 5, is constructed as a more complex neural network, following a commonly adopted architectural pattern in DDPG implementations. Its primary role is to estimate the Q-value, representing the expected return from a given state-action pair. The network receives a total of three state inputs and one action input. These inputs are concatenated and then processed through the network's layers to ultimately output a single scalar Q-value.

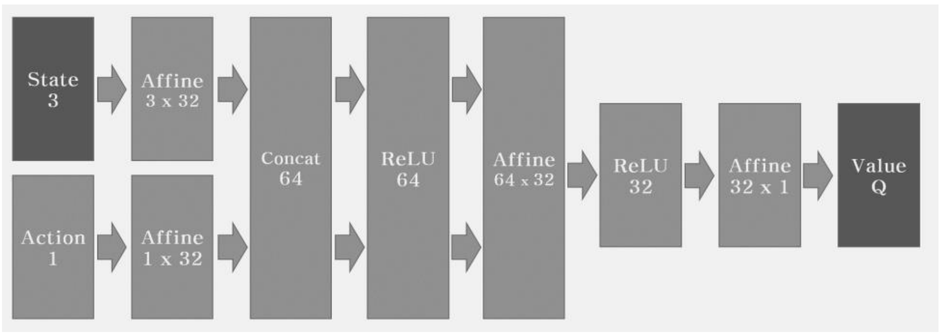


Figure 5. Critic network

The internal structure of the Critic network is composed of alternating Affine layers and ReLU activation functions. This sequence of layers and non-linearities is deliberately chosen to enable the network to model highly complex and non-linear function approximations. The Affine layers facilitate linear transformations of the input features, while the ReLU activations introduce non-linearity, allowing the network to capture intricate relationships between states, actions, and their corresponding Q-values. This design ensures that the Critic can accurately estimate the value function even in environments with highly complex dynamics and reward landscapes, thereby providing robust feedback for the Actor's policy updates.

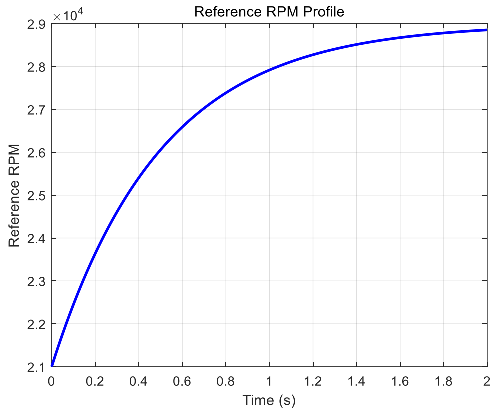


Figure 6. Examples of standard speed used in learning

Figure 6 illustrates the Reference RPM Profile employed during the training of our DDPG agent for the propulsion system. This reference signal is not static. rather, it represents the target profile that the propulsion system is designed to track and learn during each training iteration.

The specific profile is characterized by an exponential increase from an initial 21,000 RPM, asymptotically approaching 28,000 RPM. This dynamic reference is continuously provided to the propulsion system throughout the reinforcement learning process. The DDPG agent's objective is to achieve real-time optimization by continuously learning to follow this reference trajectory as optimally as possible. This iterative tracking and learning process allows the agent to discover control policies that can effectively manage the propulsion system's dynamics under varying conditions.

The repeated application of this profile throughout the training iterations is crucial for the agent to generalize and robustly track dynamic setpoints. When considering the application of such a learning methodology to real-world propulsion systems, it is highly beneficial to train the reinforcement learning agent using data derived from formalized and well-defined operational sequences, such as an aircraft's take-off sequence. Utilizing data from such structured scenarios provides a rich and relevant training environment, ensuring that the learned control policies are robust, safe, and effective for critical operational phases. This approach enables the agent to acquire expertise in managing complex transient behaviors inherent in propulsion systems during specific mission segments.

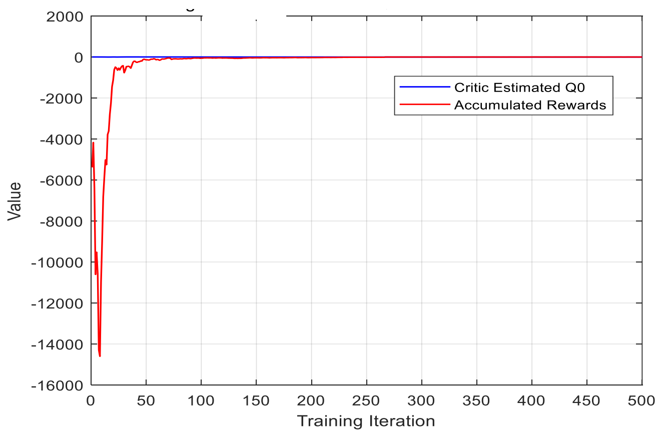


Figure 7. Critic network

Figure 7 presents a critical analysis of the DDPG agent's learning convergence by comparing the Critic network's estimated Q0 value with the Accumulated Reward obtained directly from the actual system simulation over 500 training iterations.

A fundamental indicator of successful learning in DDPG is the convergence and alignment between the estimated Q0 from the Critic network and the actual accumulated rewards experienced by the agent in the simulation. As learning progresses, a well-trained Critic should accurately predict the future accumulated rewards that the Actor's policy will achieve.

As depicted in Figure 7 the Critic's estimated Q0 values gradually converge and align closely with the Accumulated Rewards from the real system simulation. Both curves demonstrate a clear trend of maximization and subsequent convergence towards a stable, high value. This consistent agreement and convergence to a maximized reward indicate that the DDPG agent has successfully completed its learning process, acquiring an optimal policy that consistently yields high cumulative returns, and that the Critic is effectively estimating the value of this policy.

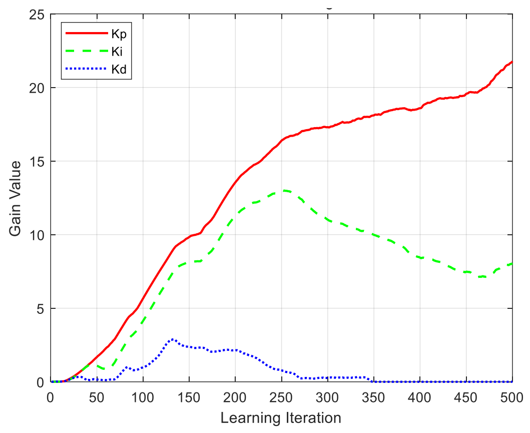


Figure 8. Training graph of PID parameters

Figure 8 illustrates the dynamic evolution of the three internal weight parameters wp,wi,wd within the Actor network over the 500 training iterations. These parameters are designed to exhibit characteristics analogous to the proportional, integral, and derivative gains of a conventional PID controller.

A particularly noteworthy observation from Figure 8 is the distinct trend of the wD parameter converging towards zero as the training progresses. This phenomenon is highly appropriate and indicative of an optimally learned control strategy. The primary reason for this behavior lies in the inherent nature of our Reference RPM Profile, which fundamentally represents a velocity-like command for the propulsion system. When the control objective is to track a velocity reference, the derivative component becomes less critical, or even detrimental, as it can introduce unnecessary oscillations or sensitivity to noise when the target is already a rate.

In systems where the reference signal is already a rate or velocity, a well-optimized controller naturally prioritizes the proportional and integral terms to achieve accurate tracking and eliminate steady-state errors. Therefore, the DDPG agent's decision to minimize the influence of the derivative term. This convergence further underscores the intelligence and efficiency of the reinforcement learning process in adapting the control policy to the specific characteristics of the reference signal and the propulsion system dynamics, thereby achieving stable and effective control without over-reliance on a less relevant control component.

CONCLUSIONS

In this study, the real-time performance optimization capability of the DDPG algorithm was rigorously validated through extensive simulations, demonstrating its superiority when compared against a conventional PID controller. These compelling simulation results affirm that the DDPG algorithm represents an effective approach for managing complex propulsion systems.
A noteworthy observation from the learning process, specifically highlighted in Figure 8, was the convergence of the derivative term towards zero. This convergence is a rational and expected phenomenon, particularly given that the propulsion system's RPM inherently represents a state of velocity. In such a context, an optimized policy would naturally minimize reliance on the derivative component as the system stabilizes and tracks the reference, thus eliminating overshoot and oscillations. Therefore, this convergence serves as a strong indicator of successful and stable learning.

The methodology proposed in this research holds significant promise as a viable solution for enabling continuous optimization within complex model environments. We anticipate that this approach can lead to more adaptive, robust, and efficient control strategies for propulsion systems and similar dynamic systems where real-time adaptability is paramount.

6. Patents

Acknowledgments: This work was supported by the Hanseo RISE Project Group, funded by the Ministry of Education and Chungcheongnam-do Province in 2025, as part of the Regional Innovation System & Education (RISE) program (2025-RISE-12-023).

REFERENCES

1. Jung, B.I.; Ji, M.S. Fuel Flow Control of Turbojet Engine Using the Fuzzy PI+D Controller. J. Adv. Navig. Technol. 2011, 15, 449–455.
2. Kim, D.K.; Hong, K.Y.; An, D.M.; Hong, S.B.; Ji, M.S. Control of UAV Turbojet Engine using Artificial Neural Network PID. J. Adv. Navig. Technol. 2014, 18, 107–113.

3. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. In NIPS Deep Learning Workshop, Lake Tahoe, NV, USA, 2013.
4. Yang, J.; Peng, W. Research on Cooperative Adaptive Cruise Control Systems Using Deep Learning-Based PID Control Techniques. Electronics 2021, 10, 2689.
5. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. In Proceedings of the International Conference on Learning Representations (ICLR), San Juan, Puerto Rico, 2–4 May 2016; pp. 1–14.
6. Bandong, S.; Nazaruddin, Y.Y.; Joelianto, E. DDPG-Based PID Optimal Controller for Position and Sway Angle of RTGC. In Proceedings of the 2023 23rd International Conference on Control, Automation and Systems (ICCAS), Yeosu, South Korea, 17-20 October 2023; pp. 696–701.
7. Jaw, L.C.; Mattingly, J.D. Aircraft Engine Controls: Design, System Analysis, and Health Monitoring; American Institute of Aeronautics and Astronautics: Reston, VA, USA, 2009.
8. Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.