JOURNAL OF MECHANICAL ENGINEERING AND SCIENCES
ISSN: 2289-4659 e-ISSN: 2231-8380
VOLUME 19, ISSUE 3, 2025, 10739 - 10755
DOI: https://doi.org/10.15282/jmes.19.3.2025.4.0842

UNIVERSITI MALAYSIA PAHANG AL-SULTAN ABDULLAH

RESEARCH ARTICLE

Development of an intelligent jet engine controller using a model-based deep deterministic policy gradient technique

E. Mohammad, M. Jahromi*, J. Pirkandi, M. Khazaee, M. Mahmoodi

Faculty of Aerospace, Malek Ashtar University of Technology, 83154/115, Iran
Phone: +989123935744, Fax: 02122985689

ABSTRACT - The rapid advancement of artificial intelligence has the potential to significantly enhance the aerospace industry, particularly through the development of intelligent engine control systems. This study aims to address the challenges of controlling complex, nonlinear aero-engines by employing Deep Reinforcement Learning techniques. Specifically, the Deep Deterministic Policy Gradient (DDPG) algorithm, within an actor-critic framework, is employed to design an adaptive controller for the nonlinear thermodynamic model of the J85 jet engine. The proposed method is evaluated against traditional PI controllers under various operating conditions, including different altitudes, Mach numbers, and humidity levels. Simulation results reveal that the DDPG-based controller outperforms PI control by achieving faster response times, 1.75 seconds (7.18%) faster during acceleration and 0.55 seconds (1.24%) during deceleration in standard conditions, and 1.09 seconds (4.79%) and 3.44 seconds (7.13%), respectively, under altered conditions. Moreover, the DDPG controller reduces the turbine inlet temperature by up to 44.97% in the first case and 38.21% in the second case, and decreases the surge margin by 54.83% and 56.18%, respectively, ensuring safer operation within the limits. These findings demonstrate the DDPG algorithm's potential for substantial improvements in engine control performance and safety. The study underscores the transformative potential of AI-driven control systems in aerospace applications, paving the way for more intelligent and adaptable engine management solutions.

ARTICLE HISTORY
Received : 28th Jan. 2025
Revised : 23rd May 2025
Accepted : 11th June 2025
Published : 30th Sept. 2025

KEYWORDS
Artificial intelligence
Aero engines
DDPG algorithm
Deep reinforcement learning

1. INTRODUCTION

Aero-engines are intricate systems requiring advanced control mechanisms to address their nonlinear nature, where even small input adjustments can lead to substantial output differences. Over time, wear and environmental influences introduce variability, making adaptive control strategies essential. Additionally, their performance is very responsive to external factors such as altitude and temperature, which directly impact efficiency. Effective control systems must handle numerous interdependent variables using advanced algorithms for optimal functionality. Historically, aircraft propulsion systems have relied heavily on PI control due to its simplicity and robustness. However, as advancements in aero-engine technology progress, there is an increasing expectation for these systems to exhibit more complex control dynamics, which drives the need for enhanced control methods to optimize performance. Over the past few years, researchers have proposed diverse superior control techniques for aero-engines, including the Linear Quadratic Regulator (LQR) and H∞ methods [1, 2]. However, the majority of these methods are designed for linear models. Researchers have also examined nonlinear methodologies to address the inherent challenges of nonlinearity. This includes (i) Gain scheduling control, which adjusts controller parameters based on operating conditions [3, 4] (ii) Model predictive control that employs a system representation to forecast upcoming performance and enhance control actions [5] (iii) Sliding mode variable structure control, which ensures robust performance by forcing system states to slide along a predetermined trajectory [6]. While implementing effective control strategies traditionally requires accurate system models and precise controllers, challenging due to the complex aerothermodynamics and dynamics of aircraft propulsion systems, this has driven increased interest in model-free, AI-based algorithms like reinforcement learning, which leverage large datasets, fast simulators, advanced neural networks, and high-performance computing to address the complexities of unknown nonlinear systems, demonstrating scalability, efficiency, and adaptability in stochastic, dynamic environments far beyond the capabilities of classical decision-making methods [7].

Through trial and error, reinforcement learning, often abbreviated as RL, is a machine learning approach focused on maximizing total rewards by discovering the optimal actions. It learns from direct interactions and is model-free, indicating that it doesn't rely on a predefined environment model. The foundation of reinforcement learning is optimal control theory, which seeks to determine the optimal policies for dynamic systems, and dynamic programming, which breaks down complex problems into simpler ones to facilitate their solution. By combining these frameworks, RL can gradually formulate efficient decision-making methods [8]. A widely utilized technique in reinforcement learning is Q-learning [9], which simplifies both action and state spaces by discretizing them and addressing problems through a table-based approach. This approach has been extensively researched and refined, resulting in variations such as SARSA, DQN,

*CORRESPONDING AUTHOR | M. Jahromi | ✉ mjahromi@mut.ac.ir

and Double DQN [10-12]. The DQN (Deep Q-Learning) technique revolutionizes the application of RL to complex problems by leveraging the power of deep learning to enhance its effectiveness in handling tasks that involve visual data, resulting in impressive performance in areas such as video game playing and robotic control. It has shown encouraging results in problems that include making decisions with discrete actions; however, its generalization to high-dimensional continuous action spaces is still under investigation [10]. Discretizing continuous action spaces can lead to a staggering increase in dimensionality, prompting researchers to introduce the Actor-Critic (AC) framework [13], a compelling alternative that effectively navigates this issue. This innovative approach combines value-based and policy-based methods, demonstrating its versatility in various applications. Whether it's controlling aircraft, managing robotic systems, steering UAVs through complex pathways, or actively damping vibrations, the AC framework has shown its strength multiple times [14-17]. The system dynamically assesses environmental feedback within the AC architecture, selecting the most appropriate control actions diligently. The actor steps in to generate actions, while the critic plays a crucial role in evaluating their quality, offering valuable feedback that sharpens control accuracy with each iteration. Currently, a vast array of Deep Reinforcement Learning algorithms, which are fundamentally based on the Actor-Critic approach, are skillfully designed to function with continuous action environments. Prominent examples of these include the Deep Deterministic Policy Gradient, referred to as DDPG [18], Twin-Delayed DDPG, known as TD3 [19], and Soft Actor-Critic, abbreviated as SAC [20]. Table 1 presents a comparative analysis of the three methods.

Table 1. Comparison between DDPG, TD3, and SAC methods

Aspect	DDPG	TD3	SAC
Policy Type	Deterministic	Deterministic	Stochastic
Main Innovations	Basic actor-critic	Twin Q, delayed updates, policy smoothing	Maximum entropy, stochastic policy
Stability	Moderate	High	High
Sample Efficiency	Moderate	Good	Very good
Complexity	Moderate	Slightly higher	Higher due to entropy components
Use Cases	Continuous control, early exploration	Continuous control, stable training	Complex environments require exploration

DDPG is a straightforward and easy-to-implement method that effectively handles continuous control tasks using deterministic policies for reliable performance in complex environments. Consequently, in this research, DDPG has been chosen as the foundational algorithm for the reinforcement learning controller. Recent advancements in reinforcement learning (RL) have significantly contributed to the evolution of aircraft engine management systems, providing robust optimization, diagnostics, and control solutions. The initial application of deep RL approaches—such as deep-Q networks and soft actor-critic algorithms—enables the multi-objective and constrained optimization of engine performance metrics, including fuel efficiency, thrust, and engine wear, across the full flight envelope [21-24]. Virtual simulation environments further facilitate self-learning and environment-aware optimization, enhancing operational efficiency and reducing maintenance costs through collaborative multi-agent frameworks [25]. Complementing these developments, RL has also advanced the diagnostics of aero-engines. Techniques such as continual contrastive learning and adaptive Kalman filters enhance fault detection and engine condition monitoring, particularly during periods of high imbalance or sudden operational changes [26-28]. For controlling aircraft engines, which is the focus of this investigation, many reinforcement learning algorithms have been employed to improve engine performance and increase reliability; however, the field still requires further development.

Among the most important works in developing this field, for example, Zhang proposed an optimal controller using reinforcement learning neural networks, specifically designed to operate at steady-state conditions for aero engines [29]. Similarly, Zheng suggested employing deep Q-learning to optimize the acceleration characteristics of aero-engines [30]. Their research suggests pinpointing the action that offers the greatest value at every stage, as the DQN method is not applicable in continuous action settings. This approach adds complexity to achieving real-time control. In contrast, a transient controller design using the DDPG algorithm was proposed in [31], with simulations confirming its effectiveness in managing rapid acceleration and deceleration of turbofan engines while ensuring system performance. Using deep reinforcement learning, Qian established a polynomial state-space mathematical model for a turbofan engine and demonstrated substantial performance enhancements through an intelligent controller developed via the DDPG algorithm [32]. Tao concentrated on finding a solution to the optimization control issue of variable cycle engines, which are made to modify their operation for increased efficiency in various scenarios [33].

Zhu developed a thrust estimation model for a turbofan engine utilizing a long-short-term memory, implementing the PPO technique (proximal policy optimization) to facilitate immediate thrust regulation [34]. Zhu introduced a virtual control system for aircraft engines that relies on reinforcement learning, utilizing long short-term memory (LSTM) and deep deterministic policy gradient (DDPG) algorithms. This approach demonstrated enhanced effectiveness compared to standard PID controllers, resulting in less overshoot and quicker responses [35]. Zhu then employed the TD3 algorithm to create an intelligent controller for the JT9D turbofan engine. This improved tracking control reduces overshoot and

provides quicker reactions [36]. To enhance the efficacy of the reinforcement learning method, Gao investigated additional uses by leveraging deep reinforcement learning, specifically focusing on the PPO technique. He also retrieved dynamic transition features from the DRL replay buffer to train the deep neural network designed for developing the prediction model [37]. Zhang presented a method for active surge control in compressors used in aero engines, leveraging deep reinforcement learning to tackle uncertainties and delays in actuators. This controller employs an Enhanced Soft Actor-Critic algorithm, exhibiting superior performance in terms of tracking precision and resilience compared to conventional approaches, thereby ensuring stable compressor functionality under various conditions [38]. Zhu presented an innovative self-evolution direct thrust control system for turbofan engines, integrating a model that predicts states while accounting for uncertainties with reinforcement learning techniques to cater to the unique variations between engines [39]. Chen introduced a data-driven method for adaptive dynamic programming to tackle thrust tracking management with unidentified dynamics in a turbofan engine. This approach reformulated the challenge into an optimal control structure and validated its efficacy using simulations [40].

Prior research has primarily focused on managing aircraft engines under stable conditions or specific operational scenarios, such as flying at a fixed speed and altitude while following commands for either speed increase or decrease. In contrast, this study proposes a comprehensive control method that addresses steady-state and transient behaviors across numerous operating conditions, including speed, altitude, and relative humidity variations. Our approach emphasizes a critical reference command that captures the engine's most nonlinear processes — namely, acceleration and deceleration — within a unified control framework.

2. METHODS AND MATERIALS

2.1 Thermodynamic Model of Engine

This research focuses on the J-85 turbojet engine, a well-known compact single-shaft turbojet created by General Electric. The J-85 engine series has been in use for over five decades due to its reliability and superior thrust-to-weight ratio. It consists of an air intake, an eight-stage axial-flow compressor, a combustion chamber, two turbine stages, and a converging exhaust nozzle. A schematic illustration of this engine is shown in Figure 1 [41].

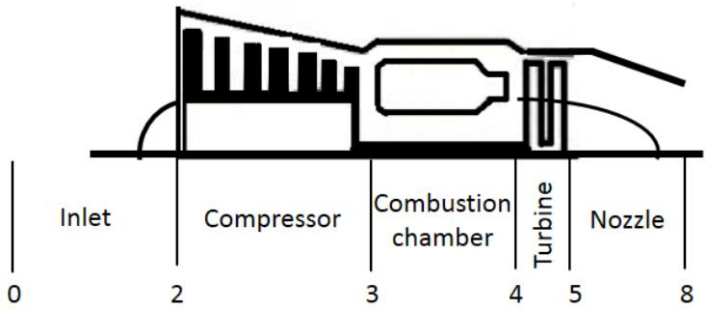

Figure 1. Schematic of the J-85 engine [41]

MATLAB Simulink has been used to construct the fundamental elements of the engine, including the air inlet, compressor, combustion chamber, turbine, and jet nozzle. Our engine model is developed using the Inter-Component Volume (ICV) method, as described by [42], wherein each component is characterized by steady-state properties and accompanied by a volume for storing mass and energy, as shown in Figure 2.

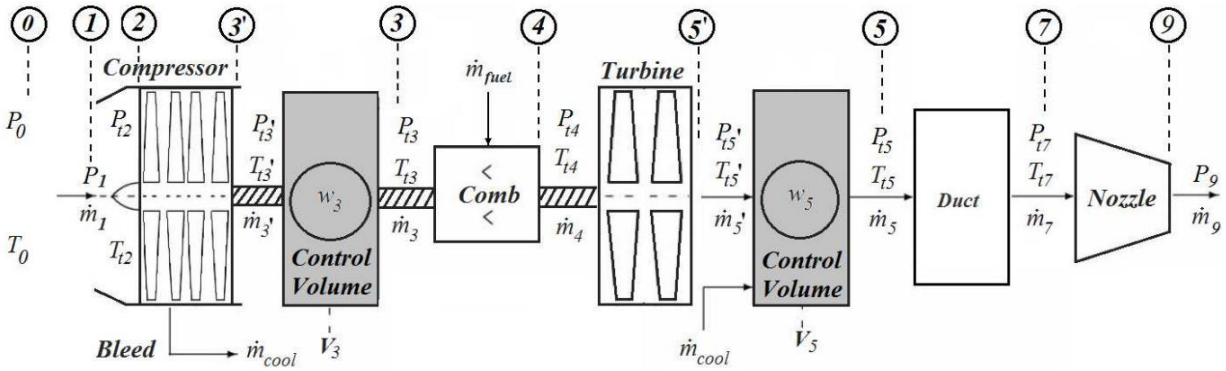

Figure 2. Turbojet engine dynamic model with the inter-component volume method

The compressor's primary performance characteristics were derived from the compressor performance maps using the Smooth C software developed by J. Kurzke [43]. Similarly, Smooth T software was utilized to obtain the main performance characteristics of the turbine from turbine performance maps [44]. The data acquired from both programs is

then utilized to finalize the development of the thermodynamic model in Simulink. To facilitate engine modeling, Table 2 presents the fundamental parameters of the engine [45].

Table 2. J85 engine design parameters

Parameter	Symbol	Value
Reference temperature (K)	T_{ref}	288.15
Reference pressure (Pa)	P_{ref}	101325
Compressor pressure ratio	P_3/P_2	6.920
Compressor efficiency (%)	$\eta_{Is\,C}$	82.5
Turbine efficiency (%)	$\eta_{Is\,t}$	88
Combustion efficiency (%)	η_b	99.5
Combustion pressure loss (%)	ΔP_{loss}	4
Mechanical efficiency (%)	η_m	99
Mass flow (kg/s)	m	19.9
Rotational speed (RPM)	N	16540
Fuel heating value (MJ/kg)	LHV	43.031
Polar moment of inertia of the shaft (kg·m^2)	I	0.7578
Turbine pressure ratio	P_5/P_4	2.5017
The effective cross-sectional area of the nozzle (m^2)	A_{nz}	0.060618
Thrust (N)	Th	14392

The analysis begins with determining the air static temperature and pressure at various altitudes using the ISA model. The values indicating the stagnation of air temperature and pressure are obtained from the ambient temperature T_{amb} and pressure P_{amb} using these equations:

$$T_0 = T_{amb} * \left[1 + \left(\frac{\gamma - 1}{2}\right) * M^2\right] \tag{1}$$

$$P_0 = P_{amb} * \left[1 + \left(\frac{\gamma - 1}{2}\right) * M^2\right]^{\frac{\gamma}{\gamma-1}} \tag{2}$$

where γ is the ratio of specific heats and M is the Mach number.

Assuming negligible heat exchange and friction between the air and the walls of the intake, the temperature and pressure at the air intake's exit can be derived from the following equations:

$$T_2 = T_0 \tag{3}$$

$$P_2 = \eta_I * P_0 \tag{4}$$

η_I is the intake efficiency.

Variations in humidity at the intake can impact the performance parameters of engines, similar to the effects observed under varying atmospheric conditions, such as pressure and temperature. This influence is primarily due to changes in the thermodynamic properties (specific heat at constant pressure, c_p, gas constant, R, and specific heat ratio, γ) of the working medium within engines, which consists of dry air, indicated by the subscript (A), and water vapor, denoted by the subscript (w), when humidity is introduced. Consequently, this interaction modifies interrelated thermodynamic variables that determine engine performance. The adjustments to these properties are mathematically delineated by specific equations referenced in [46].

Specific heat factor for moist air:

$$c_{pfac} = \frac{\left(WAR.molar * c_{pw} + (1 - WAR.molar) * c_{pA}\right)}{c_{pA}} \tag{5}$$

WAR is the ratio of water to dry air, by mass.

The gas constant factor for moist air:

$$R_{fac} = \frac{R_0}{(MW * R_A)} \tag{6}$$

$$MW = \frac{1}{\left(\left(\frac{WAR}{18.015}\right) + \left(\frac{(1 - WAR)}{28.96}\right)\right)} \tag{7}$$

R_0 is the universal gas constant

Gamma factor for moist air:

$$\gamma_{fac} = \frac{(WAR.molar * \gamma_w + (1 - WAR.molar) * \gamma_A)}{\gamma_A} \tag{8}$$

$$WAR.molar = WAR * \frac{28.96}{18.015} \tag{9}$$

To precisely determine the primary performance parameters of the compressor, it is essential to utilize performance maps specific to the J85 engine. These maps are instrumental in our analysis (sourced from the reference [45] and the GasTurb software). One map provides details regarding the corrected flow rate and pressure ratio at given corrected speeds; another map plots efficiency against pressure ratio, adjusted for speed. For an accurate depiction of the compressor's performance characteristics, we employ the Smooth C program. This utility facilitates calculations of isentropic efficiency and corrected mass flow rate as functions of both pressure ratio and corrected rotor speed, leveraging data extrapolated from these performance maps. Subsequently, using this computed data on corrected mass flow rates $m_{3'corr}$ and compressor efficiency values allow for further calculation of actual mass flows and temperatures based on established formulas using the following equations:

$$\dot{m}_{3'} = \frac{\dot{m}_{3'corr} * \frac{P_2}{P_{ref}}}{\sqrt{\frac{T_2}{T_{ref}}}} \tag{10}$$

$$T_{3'} = T_2 * \left\{1 + \frac{1}{\eta_{Is\,C}} * \left[\left(\frac{P_3}{P_2}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]\right\} \tag{11}$$

The work performed by the compressor can be quantified utilizing the following formula:

$$PW_c = \dot{m}_{3'} * c_p * (T_3 - T_2) \tag{12}$$

When modeling a compressor for a transient process, a volume is added behind the compressor through which mass and energy are accumulated. This volume is modeled by calculating the air mass W_3, pressure P_3 and temperature T_3, at its outlet using the following equations:

$$\frac{dW_3}{dt} = \dot{m}_{3'} - \dot{m}_3 - \dot{m}_{Bleed} \tag{13}$$

$$\frac{dP_3}{dt} = \frac{(\dot{m}_{3'} - \dot{m}_3 - \dot{m}_{Bleed}) * R * T_3}{V_c} \tag{14}$$

$$\frac{dT_3}{dt} = \frac{\left(\dot{m} * c_p * T\right)_{in} - \left(\dot{m} * c_p * T\right)_{out}}{W_3 * c_v} \tag{15}$$

$\dot{m}_{Bleed}$ is the compressor bleed flow rate, V_c is the compressor volume, and c_v represents the specific heat at constant volume.

In the practical combustion process, there is a decrease in pressure within the combustion chamber (ΔP_{loss}), which contrasts with theoretical models where pressure remains consistent at the chamber inlet and outlet. Consequently, to determine the pressure at the outlet of the combustion chamber, it will be calculated using the following equation:

$$P_4 = P_3 * (1 - \Delta P_{loss}) \tag{16}$$

The mass flow rate from the combustor is:

$$\dot{m}_4 = \dot{m}_3 + \dot{m}_f \tag{17}$$

$$W_4 = \frac{P_4 * V_b}{R * T_4} \tag{18}$$

V_b is the combustor volume, and $\dot{m}_f$ is the fuel flow rate.

The increase in temperature within the combustor can be determined using the formula:

$$\frac{dT_4}{dt} = \frac{1}{c_{v4} * W_4} * \left[\left(LHV * \eta_b * \dot{m}_f\right) + \left(\dot{m}_3 * c_{p3} * T_3\right) - \left(\dot{m}_4 * c_{p4} * T_4\right)\right] \tag{19}$$

To accurately determine the primary performance parameters of the turbine, it is essential to reference the turbine performance maps specific to the J85 engine. These maps have been sourced from [45] and analyzed using GasTurb software. In this analysis, we utilize Smooth T to represent turbine map data. Likewise, employing a similar methodology as Smooth C used for compressors allows us to calculate both mass flow rate and temperature through derived equations:

$$\dot{m}_{5'} = \frac{m_{5'corr} * \frac{P_4}{P_{ref}}}{\sqrt{\frac{T_4}{T_{ref}}}} \tag{20}$$

$$T_{5'} = T_4 * \left\{1 + \eta_{Is\,t} * \left[1 - \left(\frac{P_5}{P_4}\right)^{\frac{\gamma_b - 1}{\gamma_b}}\right]\right\} \tag{21}$$

γ_b is the gas specific heat ratio.

The calculation of the work performed on the turbine by the hot gases can be derived from the following equation:

$$PW_t = \dot{m}_{5'} * c_p * (T_5 - T_4) \tag{22}$$

When modeling a turbine in the transient process, a volume is added after the turbine through which mass and energy are accumulated. This volume is modeled by calculating the air mass W_5, pressure P_5, and temperature T_5 at its outlet using the following equations:

$$\frac{dW_5}{dt} = \dot{m}_{5'} - \dot{m}_5 + \dot{m}_{Bleed} \tag{23}$$

$$\frac{dP_5}{dt} = \frac{(\dot{m}_{5'} - \dot{m}_5 + \dot{m}_{Bleed}) * R * T_5}{V_t} \tag{24}$$

$$\frac{dT_5}{dt} = \frac{\left(\dot{m} * c_p * T\right)_{in} - \left(\dot{m} * c_p * T\right)_{out}}{W_5 * c_v} \tag{25}$$

V_t is the turbine volume.

Since the nozzle is convergent in the J85 engine, we can define the critical pressure in the jet nozzle by the equation:

$$P_{cr} = \left[\frac{2}{\gamma_b + 1}\right]^{\frac{\gamma_b}{\gamma_b - 1}} * P_{inlet} \tag{26}$$

Should the back pressure exceed the critical pressure, the exit pressure at the nozzle is determined accordingly:

$$P_{exit} = P_b \tag{27}$$

The mass flow rate in the nozzle:

$$\dot{m}_5 = \frac{P_5}{\sqrt{RT_5}} A_{nz} \left[\frac{P_e}{P_5}\right]^{\frac{1}{\gamma_b}} \sqrt{\frac{2\gamma_b}{\gamma_b - 1}\left[1 - \left(\frac{P_e}{P_5}\right)^{\frac{\gamma_b - 1}{\gamma_b}}\right]} \tag{28}$$

Thrust produced:

$$Th = c_v \dot{m}_5 \sqrt{2c_p T_5 \left[1 - \left(\frac{P_e}{P_5}\right)^{\frac{\gamma_b - 1}{\gamma_b}}\right]} \tag{29}$$

Velocity at the exit of the nozzle:

$$Ve = \sqrt{\frac{2\gamma_b}{\gamma_b - 1} RT_5 \left[1 - \left(\frac{P_e}{P_5}\right)^{\frac{\gamma_b - 1}{\gamma_b}}\right]} \tag{30}$$

Conversely, if the back pressure falls below the critical threshold:

$$P_{exit} = P_{cr} \tag{31}$$

$$\dot{m}_5 = \frac{P_5}{\sqrt{RT_5}} A_{nz} \sqrt{\gamma_b \left[\frac{2}{\gamma_b + 1}\right]^{\frac{\gamma_b + 1}{\gamma_b - 1}}} \tag{32}$$

$$Th = c_v \dot{m}_5 \sqrt{2c_p T_5 \left[1 - \left(\frac{P_{cr}}{P_5}\right)^{\frac{\gamma_b - 1}{\gamma_b}}\right]} + A_{nz}(P_{cr} - P_e) \tag{33}$$

$$Ve = \sqrt{\frac{2\gamma_b}{\gamma_b - 1} RT_5 \left[1 - \left(\frac{P_e}{P_5}\right)^{\frac{\gamma_b - 1}{\gamma_b}}\right]} \tag{34}$$

Specific fuel consumption is calculated from the following equation:

$$SFC = \frac{\dot{m}_f}{Th} \tag{35}$$

In transient simulations, an imbalance arises when the work output of a turbine does not match that consumed by a compressor. Such discrepancies cause variations in speed at each time step because surplus power on the shaft results in a torque imbalance. Consequently, fluctuations in rotational speed can be assessed using this equation:

$$\frac{dN}{dt} = \left(\frac{30}{\pi}\right)^2 \frac{1}{IN} [PW_t - PW_c/\eta_m] \tag{36}$$

All of these calculations have been implemented in Simulink. Finally, the engine was modeled as shown in Figure 3.

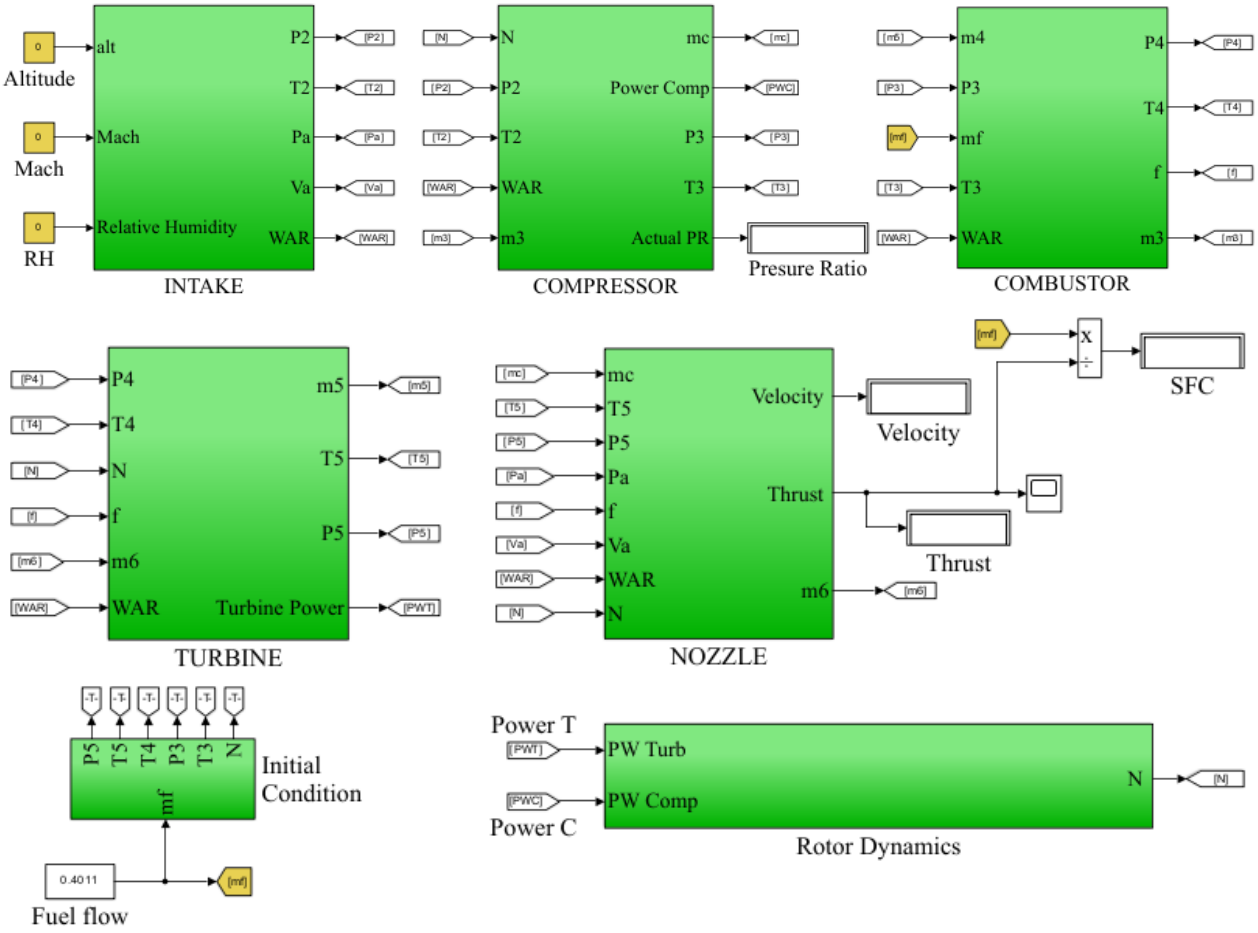

Figure 3. Thermodynamic model for J85 turbojet engine

2.2 DDPG Algorithm

The chief purpose of the agent in reinforcement learning is to enhance its policy to achieve the highest possible cumulative reward. The agent first observes the environment's current state, represented by the letter s. Based on the learned policy μ, it chooses an action, α. This decision affects the state of the environment, causing it to change and rewarding the agent in the process. The Bellman equation is used for the iterative improvement of the state-action value function, denoted as $Q^*(s, a)$ [8].

$$Q^*(s, a) = E_{\acute{s}\sim S}[r + \gamma max_{\acute{a}} Q^*(\acute{s}, \acute{a})] \tag{37}$$

In this context, whenever you see that prime notation, think of it as pointing towards the values we'll encounter at the next moment. The symbol E [·] represents this expected function, capturing our predictions about what lies ahead. Meanwhile, γ, which serves as our discount factor, subtly adjusts the value of future rewards. The reward term (r) is the function that tells us just how gratifying those outcomes can be (the reward function).

In real life, the $Q^*(s,a)$ is typically approximated by a function, meaning that $Q^*(s,a) \approx Q^*(s,a|\theta)$. One way to compute the parameter θ is to minimize the loss function. The definition of this loss function is:

$$L_i(Q_i) = E_{\acute{s}\sim S}\left[\left(y_i - Q(s,a|\theta_i)\right)^2\right] \tag{38}$$

$$y_i = E_{\acute{s}\sim S}[r + \gamma max Q(\acute{s},\acute{a}|\theta_{i-1})] \tag{39}$$

The formula for the gradient is derived by taking the derivative of the loss function's variables:

$$\nabla_{\theta_i} L_i(\theta_i) = E\left[\left(r + \gamma max_{\acute{a}} Q(\acute{s},\acute{a}|\theta_{i-1}) - Q(s,a|\theta_i)\right)\nabla_{\theta_i} Q(s,a|\theta_i)\right] \tag{40}$$

The Bellman equation must be solved to find the best course of action. Using the experience replay buffer is a crucial part of the DQN algorithm. The transition samples, those little nuggets of experience represented as (s, α, r, s ′), are gathered from the agent when it interacts with its surroundings (environment) and are carefully stored in this buffer. The learning process utilizing this strategy is made more stable by reducing the correlation between successive samples. By adding a target network, the DDPG algorithm improves the performance of the Actor-Critic network structure. The Actor and Critic neural networks are replicated to form these target networks, as shown in Figure 4, guaranteeing the same architecture and initial parameters [36].

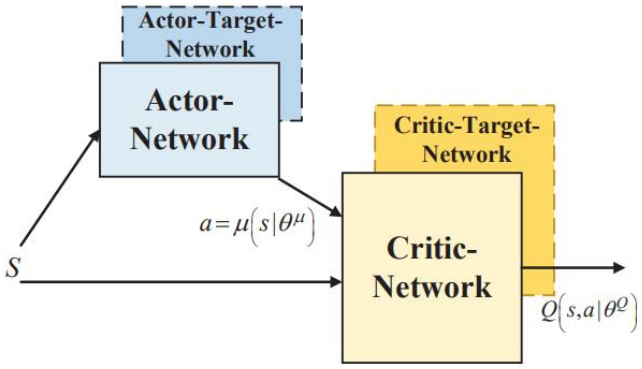

Figure 4. The structure of the DDPG networks [36]

This structure stabilizes the learning process by separating the choices of strategy, the calculation of the value function, and the parameter updates. Actor neural networks are designed to make decisions about actions based on the current state and update parameters iteratively. The actor target network determines the best course of action (α') in the meantime by examining the subsequent state sampled from the experience replay buffer, denoted as $\alpha = \mu(s|\theta^\mu)$. The goal given to the Critic network is to evaluate the existing Q-value function $Q(s,a|\theta^Q)$ and modify its parameters accordingly. Nonetheless, the Q value that serves as the target in the TD error is mainly derived from the Critic target network.

Below is the formula used to establish the target Q value:

$$y_i = r + \gamma Q\left(\acute{s},\mu\left(\acute{s}|\theta^{\acute{\mu}}\right)|\theta^{\acute{Q}}\right) \tag{41}$$

where $Q\left(\acute{s},\mu\left(\acute{s}|\theta^{\acute{\mu}}\right)|\theta^{\acute{Q}}\right)$ is the output of the Critic target network, and y_i is the target Q value. The Actor target network's production is represented by $\mu\left(\acute{s}|\theta^{\acute{\mu}}\right)$.

The TD error is minimized to train the Critic, where the square mean value of the TD error:

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q(s,a|\theta^Q)\right)^2 \tag{42}$$

$Q(s,a|\theta^Q)$ represents the Critic network output.

The policy by the Actor network is mapped onto the given state, which then refines the current policy to determine the corresponding action. The primary goal of the update procedure is to determine the optimal policy that maximizes cumulative rewards. As a result, the expectation of cumulative rewards is defined as the objective function for training the neural network. The formula that goes along with it is as follows:

$$J = E\left[\sum_{i=t}^{T}\gamma^{(i-t)} r(s_i,a_i)\right] \tag{43}$$

J is the objective function in this case.

The equation below demonstrates how the gradient of the objective function connects with the gradient of $Q(s,a|\theta^Q)$, which was confirmed by Silver in 2014 [47]:

$$\nabla_{\theta^\mu} J \approx E[\nabla_{\theta^\mu} Q(s,a|\theta^Q)|s = s_i, a = \mu(s_i|\theta^\mu)] \tag{44}$$

where, $\nabla_{\theta^\mu} Q(s, a|\theta^Q)$ is the gradient of $Q(s, a|\theta^Q)$ and $\nabla_{\theta^\mu} J$ is the gradient of the objective function.

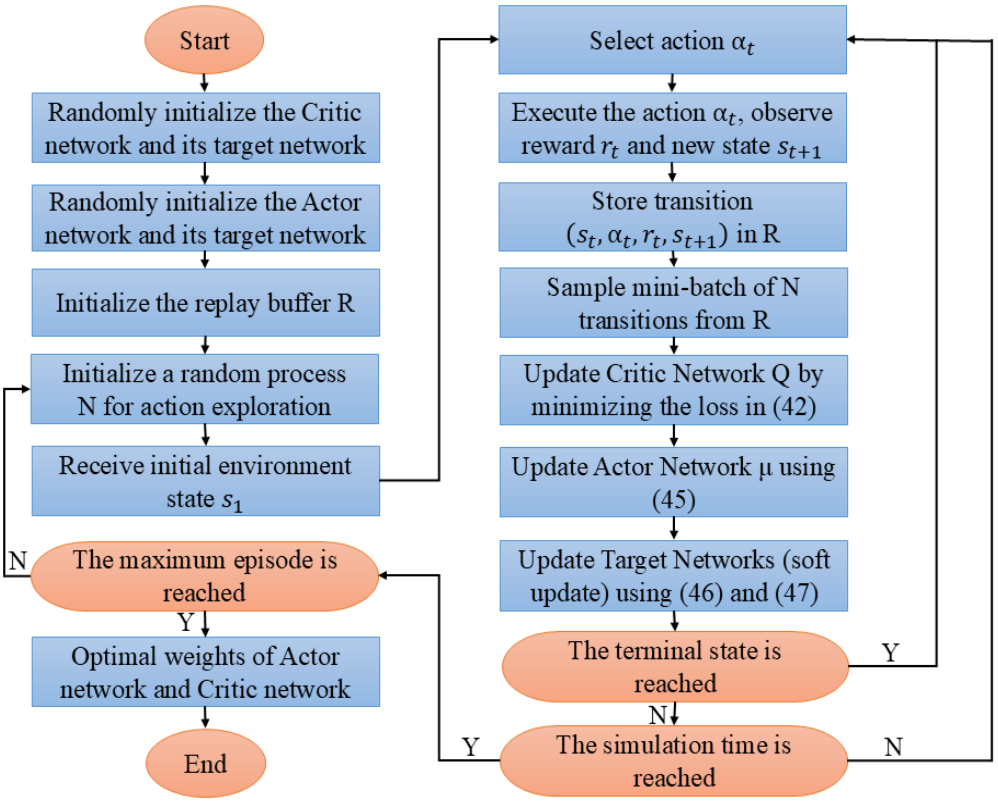

Figure 5. The training process with DDPG

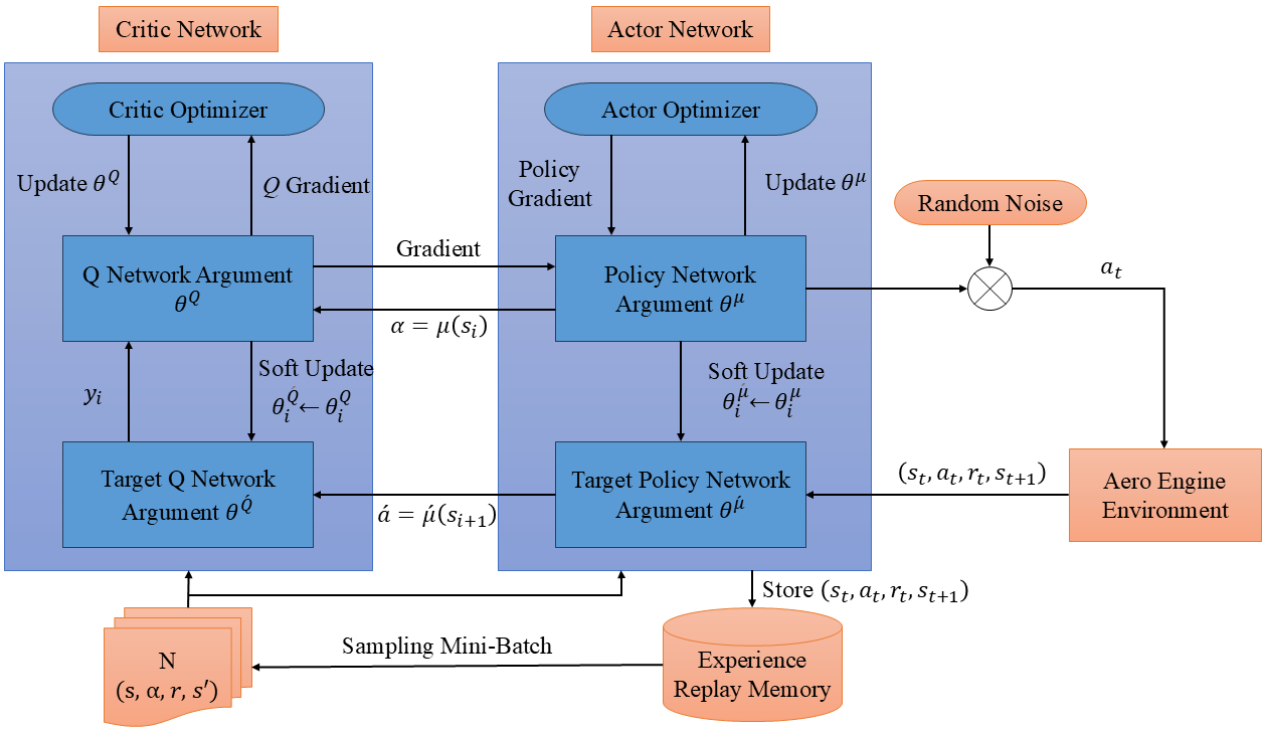

Figure 6. Flow chart of the DDPG algorithm

Batch samples are used to update the Actor neural network during training. The ultimate update formula is:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_a Q(s, a|\theta^Q)|\, s = s_i, a = \mu(s_i). \nabla_{\theta^\mu} \mu(s|\theta^\mu)|s_i \tag{45}$$

N represents the batch sample size.

The soft-update method updates the target networks. The revised formulas are as follows:

$$\theta_i^{\acute{Q}} \leftarrow \tau\theta_i^{Q} + (1-\tau)\theta_i^{\acute{Q}} \quad (46)$$

$$\theta_i^{\acute{\mu}} \leftarrow \tau\theta_i^{\mu} + (1-\tau)\theta_i^{\acute{\mu}} \quad (47)$$

where, τ is the soft update rate.

The method for training the agent using DDPG is illustrated in Figure 5.

The DDPG method relies on the Actor-Critic architecture and consistently engages with its connected environment, specifically the turbojet engine in this study, to train its network iteratively. Within this algorithm, the Actor policy network is responsible for surveying the environment and deciding on actions, where the Critic evaluation network assesses the value of each action, influencing the gradient update direction for the Actor policy network. This method, which utilizes an online network and its target for adjusting parameters, results in more consistent updates and significantly enhances the learning efficiency of the algorithm [48]. The structure of the framework and the detailed steps of the DDPG algorithm are illustrated in Figure 6.

2.3 The Jet Engine Control System's Organization

The fundamental structure of a feedback control system with closed-loop output is demonstrated in Figure 7. For implementing transient control using a reinforcement learning method, the controller segment is replaced by components for the agent, observation, and reward, while employing the designated command and control values [31]. The primary goal of this study is to ensure that, by modifying the main fuel flow, m_f, the corrected compressor rotor speed, $N_{cor,}$ precisely complies with the reference control commands while maintaining engine safety. In this setup, the jet engine controller serves as the agent, and the jet engine represents the environment, executing acceleration and deceleration processes. The replay buffer stores the set of values (s_t, a_t, r_t, s_{t+1}) generated from the connection between the jet engine and the controller at each moment. This data provides important training samples to support policy π optimization. The agent operates within the actor-critic structure, where the actor determines actions according to established rules. The critic provides the Q value function for the concurrent assessment of action quality. Relative humidity (RH), altitude (alt), Mach number (M), fuel flow (m_f), turbine inlet temperature (T4), compressor surge margin (SM), and control error (e), which is the difference between the reference and actual corrected rotor speed, are among the various flight conditions that are included in the observational inputs.

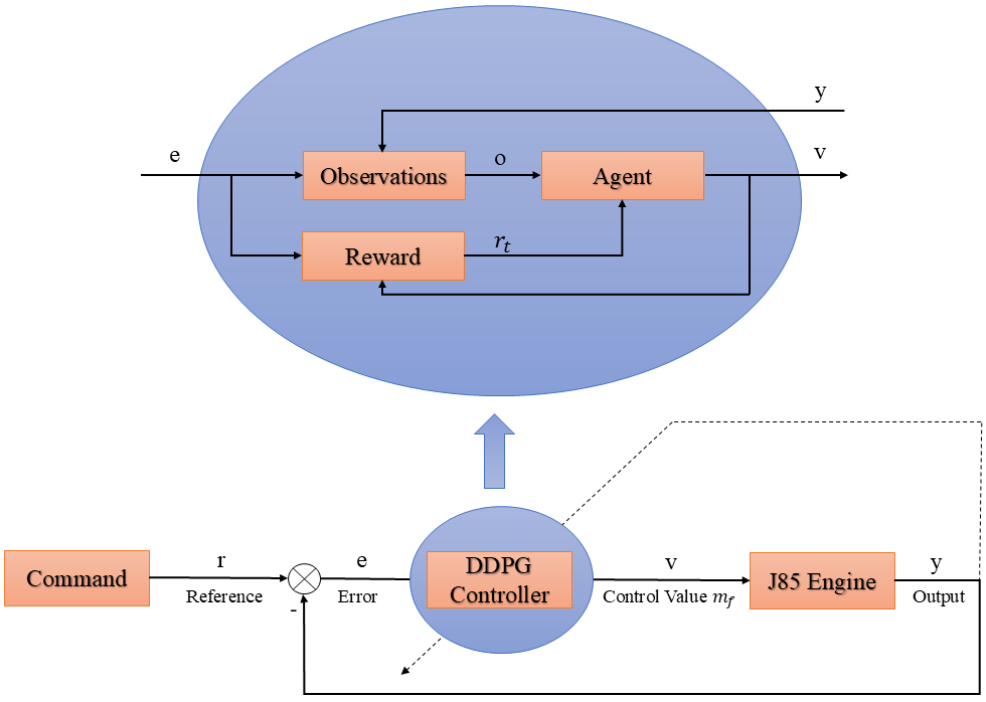

Figure 7. Structure of a conventional feedback control mechanism founded on reinforcement learning

The reward, r_t functions as a crucial signal, conveying information about the task's goals and acting as a jet engine's feedback to the critic. Creating the reward is essentially the same as deciding what the agent wants to learn. By appropriately adjusting the jet engine input (m_f) under varying flight conditions, the main goal of control is to reduce the error between the controlled variable [Ncor,t] and the reference value [Ncor,r]. Jet engine operating conditions are harsh, so it's imperative to keep important performance metrics, such as rotational speed (N), surge margin (SM), and turbine inlet temperature (T4), within safe bounds. In light of this control goal, the reward signal has the following definition:

$$r_t = -pe^2 - \begin{pmatrix} w_1[\max(N(t) - N_l(t), 0)]^2 + \\ w_2[\max(T4(t) - T4_l(t), 0)]^2 + \\ w_3[\max(SM(t) - SM_l(t), 0)]^2 \end{pmatrix} \quad (48)$$

The positive gain coefficients in this case are represented by p and w; the key performance parameters are N(t), T4(t), and SM(t), and the corresponding upper limits for those parameters are indicated by $N_l(t)$, $T4_l(t)$, and $SM_l(t)$. Equation (48), on the right-hand side, has a first error term, $-pe^2$, which ensures the controller's tracking efficiency, while the second aims to provide limited safeguarding during the process. This second term becomes zero if the key performance parameters remain within the permitted limit, ensuring that the control policy remains unaffected. Nevertheless, the second term imposes a significant negative benefit on the agent when these parameters surpass the established limit. This mechanism successfully preserves system integrity by preventing the agent from taking actions that could result in an over-limit problem.

An engine input momentum term is included to reduce abrupt changes, reducing the possibility of catastrophic events such as surges and stalls caused by erratic behavior. This equation can be used to express this term [37] :

$$u_t = \varepsilon a_t + u_{t-1} * (1 - \varepsilon) \quad (49)$$

The momentum factor in this case is represented by ε ∈ [0, 1], and the jet engine's input is indicated by u_t. This method guarantees that when ε is much less than 1, the input can remain stable even if a_t experiences sudden fluctuations. Figure 8 shows the configuration of the jet engine control system, which was implemented in Simulink/MATLAB.

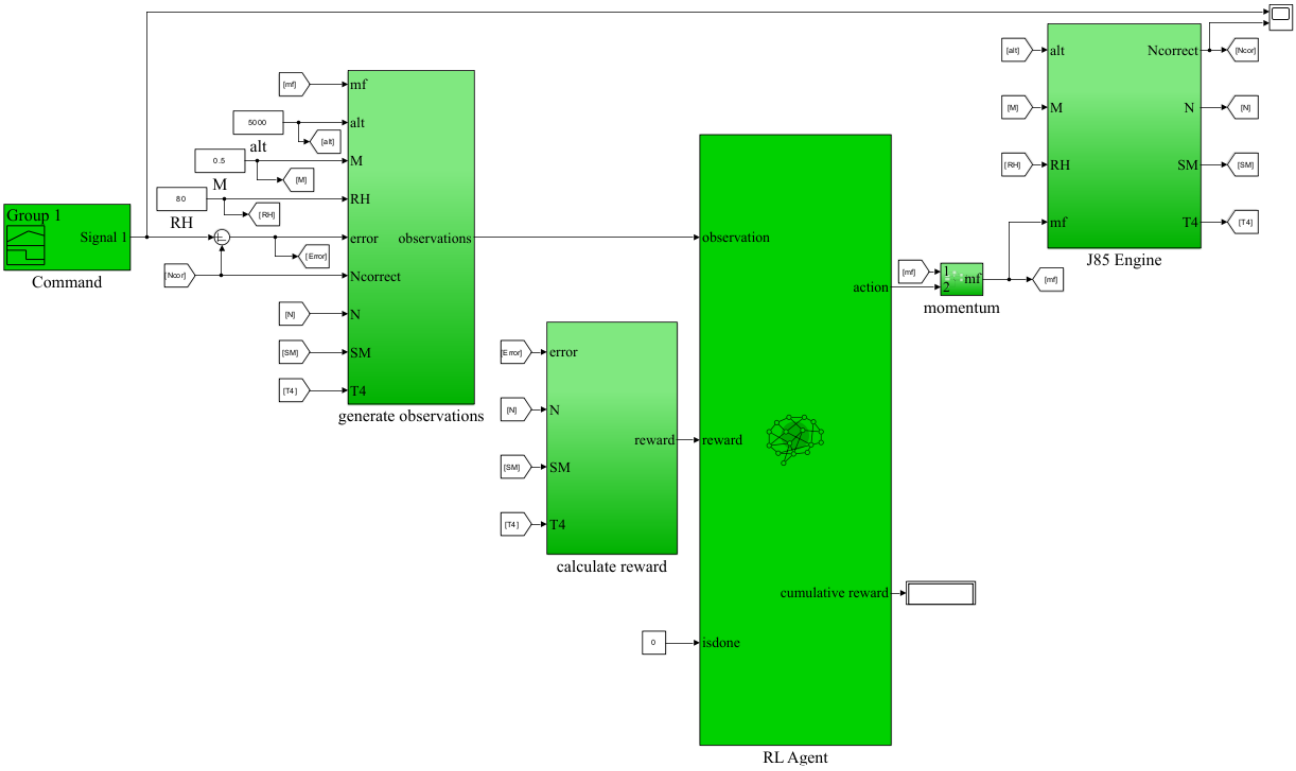

Figure 8. Deep reinforcement learning-based jet engine control system

3. RESULTS AND DISCUSSION

3.1 Training Options

Table 3 describes the parameters configured for training an Actor-Critic reinforcement learning model. The model maintains a replay buffer of 1,000,000 past experiences and trains on random mini-batches of 128 experiences per step, at the cost of taking action every 0.02 seconds to ensure real-time responsiveness. The Critic has three hidden layers, with the number of units in each layer set to 256 for approximating complexity. In contrast, the Actor has two hidden layers with 256 units to facilitate quicker action. Both parts have a gradient threshold of 1, which prevents the update from being influenced by the previous one and enables stable learning. The remaining parameters are maintained to stabilize training and promote exploration. The discount factor of 0.99 represents the reduced weight on future rewards, not just current successes. The target smooth factor of 0.003 is used to update slowly, and a momentum of 0.008 is also applied for the smoothing learning procedure in every test. The exploration policy has a controlled random factor (0.1) and decays very slowly (by 0.00001) to allow the model to try new strategies at the beginning, which is then adjusted over time to follow a more optimal strategy as it gains more experience. Together, these force parameters contribute to a well-organized training regimen that promotes sample-efficient learning and remains stable throughout the training process.

Table 3. Training parameters

Parameter	Value
Experience Buffer Length	1000000
Mini-Batch Size	128
The gradient threshold of Critic	1
The learning rate of Critic	0.001
Discount Factor	0.99
The gradient threshold of the Actor	1
The learning rate of the Actor	0.0001
Hidden layers of the Critic	3 (256 units each)
Hidden layers of the actor	2 (256 units each)
Sample time	0.02
Momentum factor	0.008
Noise Options' Variance Decay Rate	0.00001
Target Smooth Factor	0.003
Noise Options Variance	0.1

3.2 Performance of the DDPG Controller

To evaluate the effectiveness of the proposed method, we conducted comparison simulations between the PI approach with the min-max limit protection and the new method. Engine acceleration and deceleration are the most significant nonlinear processes among the transient processes. Accordingly, we considered a two-stage maneuver over sixty seconds. This maneuver involves three corrected rotor speeds, starting at a corrected rotor speed of 0.67 for 20 seconds, increasing to 1, remaining at this value for another 20 seconds, and then returning to the initial speed. In this paper, we will examine two case studies at various operating conditions, including altitude, Mach number, and relative humidity, to assess the impact of the control system. The selected scenario for each case is the worst-case scenario, involving sudden changes (rapid increase and decrease in speed) on a massive scale (from 0.67 to 1). As a result, the control system performance against this scenario is evaluated.

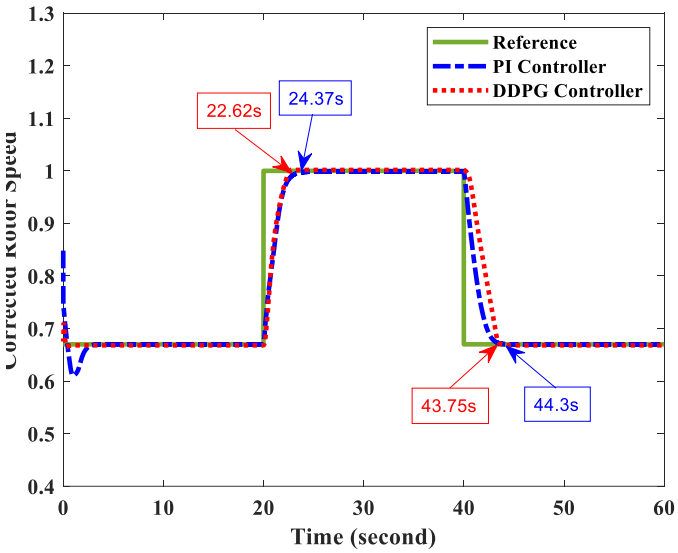

Figure 9. Corrected rotational speed response curve

3.2.1 Scenario 1: alt=0 m, M=0, RH=0 %

In this case study, the operating conditions for both methods (DDPG and PI) are at standard atmospheric conditions, specifically at altitude alt = 0 m, a Mach number of M = 0, and a relative humidity of RH = 0%. Figures 9, 10, and 11 display the simulation results for the proposed method and the PI controller. Figure 9 illustrates the response of the corrected rotor speed to the two controllers. The response speed of the jet engine with the DDPG controller is better than that of the PI controller during acceleration and deceleration. During acceleration with DDPG, the corrected rotor speed reaches the reference value at 22.62 seconds (with a steady-state error of no more than 0.3 %), whereas PI reaches the reference value at 24.37 seconds; That is, the engine rotor responds up to 7.18 % faster with the DDPG controller than with the PI controller. During deceleration, the corrected rotor speed reaches the reference value with DDPG at 43.75 seconds, whereas PI reaches the reference value at 44.3 seconds; that is, the engine rotor responds up to 1.24 % faster with the DDPG controller than with the PI controller. The figure also shows that the two controllers exhibit no overshoot

or undershoot during acceleration and deceleration. Note that there is an undershoot for the PI controller at the beginning of the simulation.

Figure 10 illustrates that the control system effectively mitigates abrupt changes in the turbine inlet during acceleration and deceleration, thereby helping to prevent issues such as combustion chamber flameouts and turbine blade damage. The presence of the DDPG controller reduces the turbine inlet temperature during acceleration by 1034 Kelvin (44.97%), while the PI controller reduces it by 1063 Kelvin (46.23%), which gives the PI an advantage in this area. Conversely, the surge margin during acceleration remains within safe working limits as shown in Figure 11. The surge margin is reduced by 0.34 (54.83%) with the DDPG controller, while it is reduced by 0.296 (47.74%) with the PI controller.

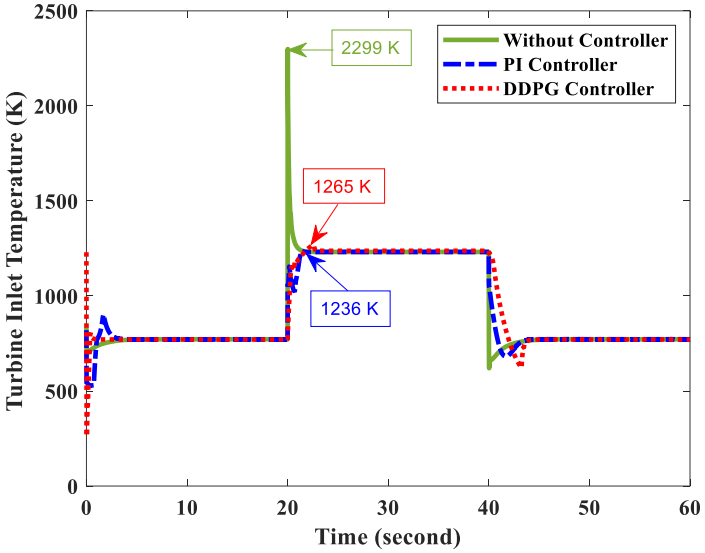

Figure 10. Turbine inlet temperature response curve

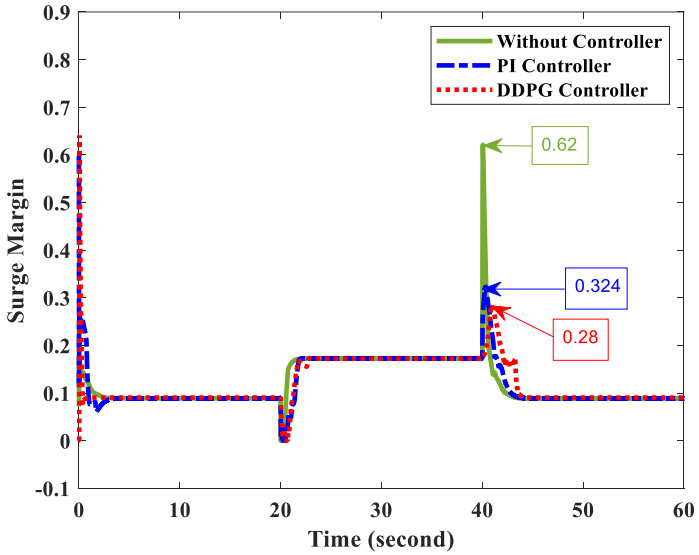

Figure 11. Compressor surge margin response curve

3.2.2 Scenario 2: alt=5000 m, M=0.5, RH=80 %

The primary purpose of this case study is to investigate the performance of the control system at different operating conditions (altitude, Mach number, and relative humidity). As mentioned in the modeling section, the integral-thermodynamic model allows simulation of the effects of altitude, Mach number, and relative humidity. In these new conditions, similar to the previous case study, we consider the worst-case scenario and a command from 0.67 to 1. Figures 12, 13, and 14 illustrate the system input command and system response, turbine inlet temperature, and surge margin (for an altitude of 5000 meters, a Mach number of 0.5, and a relative humidity of 80%). Similar to the previous case study, it can be observed that the control system successfully achieves its primary objectives of having a fast response with minimal error (Figure 12) where the DDPG and PI controller reach the reference value during acceleration in 21.65 and 22.74 seconds respectively (DDPG responds 4.79 % faster than PI with a steady-state error of no more than 0.4 %), and in 44.76 and 48.2 seconds during deceleration respectively (DDPG responds 7.13 % faster than PI). In addition, the DDPG controller prevents the engine's hot section temperature from entering the danger zone and avoids combustion chamber

flameout (Figure 13). The presence of the DDPG controller reduces the turbine inlet temperature during acceleration by 859 Kelvin (38.21%), as opposed to the PI controller, which reduces it by 725 Kelvin (32.25%). On the other hand, it maintains the surge margin within the safety limit in transient regimes (Figure 14), where the surge margin is reduced by 0.395 (56.18%) with the DDP controller, compared to a reduction of 0.348 (49.5%) with the PI controller.

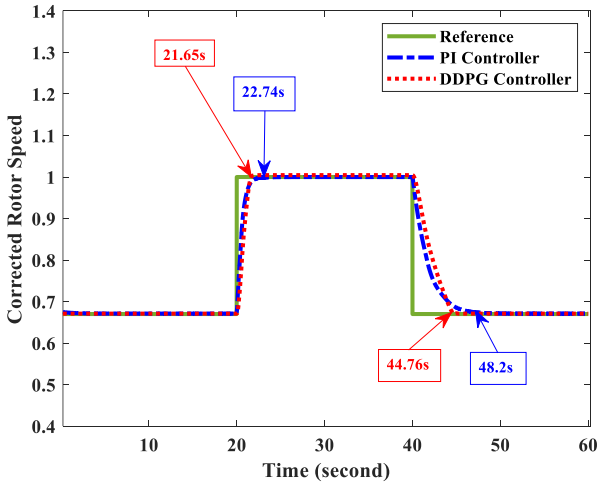

Figure 12. The response of the corrected rotor speed

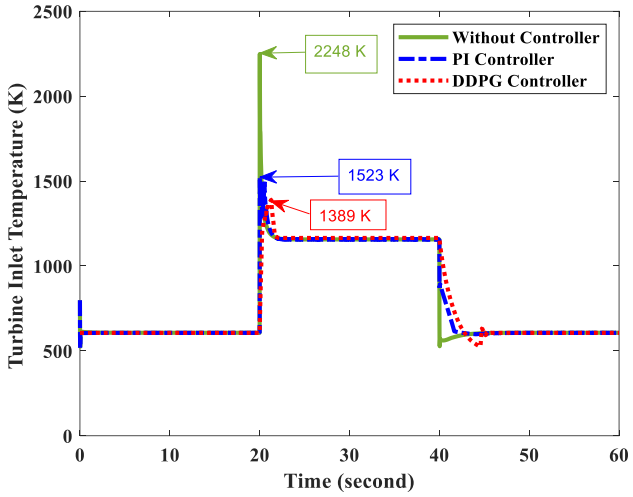

Figure 13. Turbine inlet temperature response curve

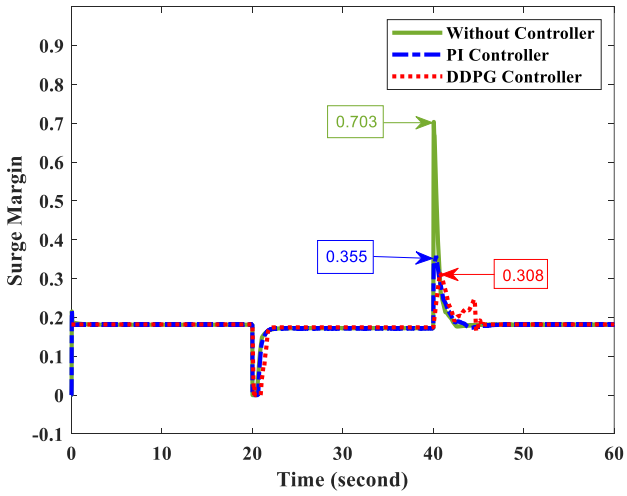

Figure 14. Compressor surge margin response curve

4. CONCLUSIONS

This paper describes a controller creation process based on reinforcement learning for jet engines. A nonlinear engine model serves as the simulation environment for the reinforcement learning process. The actor-critic framework is employed within the DDPG algorithm to create the intelligent controller. The performance of the DDPG controller was

evaluated in two cases: the first was in standard atmospheric conditions, where altitude, Mach number, and relative humidity were set to zero, and the second case was at an altitude of 5,000 meters, a Mach number of 0.5, and a relative humidity of 80%. Based on the simulation results, this paper successfully demonstrated a DDPG-based intelligent controller for jet engines, showing superior performance over PI control in simulation under varying conditions, with faster response speeds and improved engine safety. In contrast to the PI controller, the system's response speed with the DDPG controller for the first case study was 1.75 seconds (7.18 %) faster during acceleration and 0.55 seconds (1.24 %) faster during deceleration, and for the second case study was 1.09 seconds (4.79 %) faster during acceleration and 3.44 seconds (7.13 %) faster during deceleration. The turbine inlet temperature during acceleration is reduced by 44.97% in the first case and 38.21% in the second case with the DDPG controller, compared to 46.23% and 32.25% with the PI controller, thereby protecting the engine from overheating. Additionally, the surge margin is reduced by 54.83% in the first case and 56.18% in the second case with the DDPG controller, compared to 47.74% and 49.5% for a PI controller, thereby ensuring that the engine operates within safe limits. Future research should explore the real-time implementation and robustness testing of this approach. Additionally, it is possible to utilize more advanced artificial intelligence algorithms, incorporating risk assessments that can be performed during predictions.

ACKNOWLEDGEMENTS

The authors thank Malek Ashtar University of Technology, Iran, for their research facilities and support. This study was not supported by any grants from funding bodies in the public, private, or not-for-profit sectors.

CONFLICT OF INTEREST

The authors declare that they have no conflicts of interest.

AUTHORS CONTRIBUTION

E. Mohammad (Conceptualization; Formal analysis; Visualisation; Supervision)

M. Jahromi (Investigation; Data curation; Writing - original draft; Resources)

J. Pirkandi (Methodology; Writing - review & editing; Supervision)

M. Khazaee (Project administration; Supervision)

M. Mahmoodi (Project administration; Supervision)

AVAILABILITY OF DATA AND MATERIALS

The data supporting this study's findings are available on request from the corresponding author.

ETHICS STATEMENT

The authors state that no generative artificial intelligence (AI) or AI-assisted methods were used in the writing process of this work to create new content, ideas, or theories. The authors have utilized AI to enhance readability and improve language. The human tightly managed and supervised this application.

REFERENCES

[1] D. E. Miller, M. Rossi, "Simultaneous stabilization with near optimal LQR performance," *IEEE Transactions on Automatic Control*, vol. 46, no. 10, pp. 1543-1555, 2001.

[2] K. Zhu, J. Zhao, G. M. Dimirovski, "H∞ tracking control for switched LPV systems with an application to aero-engines," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 3, pp. 699-705, 2016.

[3] L. Gou, Z. Liu, D. Fan, H. Zheng, "Aeroengine robust gain-scheduling control based on performance degradation," *IEEE Access*, vol. 8, pp. 104857-104869, 2020.

[4] D. Navarro-Tapia, A. Marcos, S. Bennani, "The VEGA launcher atmospheric control problem: A case for linear parameter-varying synthesis," *Journal of the Franklin Institute*, vol. 359, no. 2, pp. 899-927, 2022.

[5] M. Montazeri-Gh, A. Rasti, A. Jafari, M. Ehteshami, "Design and implementation of MPC for turbofan engine control system," *Aerospace Science and Technology*, vol. 92, pp. 99-113, 2019.

[6] S. Liu, J. Bai, Q. Wang, W. Wang, "Tracking controller design for aero-engine based on improved multi-power reaching law of sliding mode control," *International Journal of Aeronautical and Space Sciences*, vol. 20, no. 3, pp. 722-731, 2019.

[7] P. Razzaghi, A. Tabrizian, W. Guo, S. Chen, A. Taye, E. Thompson, et al., "A survey on reinforcement learning in aviation applications," *Engineering Applications of Artificial Intelligence*, vol. 136, p. 108911, 2024.

[8] R. S. Sutton, A. G. Barto. Reinforcement Learning: An Introduction. 1st Ed. Cambridge: MIT Press Cambridge, 1998.

[9] J. Clifton, E. Laber, "Q-learning: Theory and applications," *Annual Review of Statistics and Its Application*, vol. 7, no. 1, pp. 279-301, 2020.

[10] H. Jiang, R. Gui, Z. Chen, L. Wu, J. Dang, J. Zhou, "An improved sarsa (λ) reinforcement learning algorithm for wireless communication systems," *IEEE Access*, vol. 7, pp. 115418-115427, 2019.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, 2015.

[12] H. Hasselt, "Double Q-learning," *Advances in Neural Information Processing Systems*, vol. 23, pp. 1-9, 2010.

[13] V. Konda, J. Tsitsiklis, "Actor-critic algorithms," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1-7, 1999.

[14] Y. Yuan, D. Zhou, "Reinforcement learning for dual-control aircraft six-degree-of-freedom attitude control with system uncertainty," *Aerospace*, vol. 11, no. 4, p. 281, 2024.

[15] R. Liu, F. Nageotte, P. Zanne, M. de Mathelin, B. Dresp-Langley, "Deep reinforcement learning for the control of robotic manipulation: a focussed mini-review," *Robotics*, vol. 10, no. 1, p. 22, 2021.

[16] M. Chowdhury, S. Keshmiri, "Interchangeable reinforcement-learning flight controller for fixed-wing UASs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 60, no. 2, pp. 2305-2318, 2024.

[17] Z.-C. Qiu, Y. Yang, X.-M. Zhang, "Reinforcement learning vibration control of a multi-flexible beam coupling system," *Aerospace Science and Technology*, vol. 129, p. 107801, 2022.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, et al., "Continuous control with deep reinforcement learning," *arXiv, pp. 1-14*, 2016.

[19] S. Fujimoto, H. Hoof, D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, pp. 1587-1596, 2018.

[20] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, pp. 1861-1870, 2018.

[21] J. Fang, Q. Zheng, W. Liu, H. Zhang, "Optimization control with multi-constraint of aeroengine acceleration process based on reinforcement learning," in *Third International Conference on Artificial Intelligence and Computer Engineering (ICAICE 2022)*, vol. 12610, pp. 1374-1383, 2023.

[22] Z. Wei, Z. Zhao, Z. Zhou, J. Ren, Y. Tang, R. Yan, "A deep reinforcement learning-driven multi-objective optimization and its applications on aero-engine maintenance strategy," *Journal of Manufacturing Systems*, vol. 74, pp. 316-328, 2024.

[23] P. Li, W. Tang, J. Zhu, J. Dong, "Intelligent optimization method of rotor speed control policy for turbofan engine based on soft actor-critic," in *International Conference on Advanced Unmanned Aerial Systems*, pp. 549-559, 2024.

[24] J. Yao, R. Li, Y. Guo, H. Wu, G. Cui, Z. Zhou, "Online performance seeking control of aero-engine based on stabilization-optimization dual loop," *Aerospace Science and Technology*, vol. 158, p. 109927, 2025.

[25] Z. Wei, Z. Zhao, Z. Zhou, R. Yan, "Collaborative-sequential optimization for aero-engine maintenance based on multi-agent reinforcement learning," *Expert Systems with Applications*, vol. 247, p. 123358, 2024.

[26] H. Wu, S. Zhong, M. Zhao, X. Fu, Y. Zhang, S. Fu, "Continual contrastive reinforcement learning: Towards stronger agent for environment-aware fault diagnosis of aero-engines through long-term optimization under highly imbalance scenarios," *Advanced Engineering Informatics*, vol. 65, p. 103297, 2025.

[27] R. Schirru, D. Q. Vu, S. Razakarivony, S. Thépaut, A. Bauny, "Adaptive Kalman filter by reinforcement learning for monitoring aircraft engines'performance against abrupt events," in *Turbo Expo*, pp. 1-12, 2025.

[28] J. Xu, Y. Wang, Z. Wang, X. Wang, Y. Zhao, "Transient gas path fault diagnosis of aero-engine based on domain adaptive offline reinforcement learning," *Aerospace Science and Technology*, vol. 155, p. 109701, 2024.

[29] H. Zhang, S. Wei, G. Xu, "Steady state controller design for aero-engine based on reinforcement learning NNs," in *29th Chinese Control And Decision Conference (CCDC)*, pp. 2168-2173, 2017.

[30] Q. Zheng, C. Jin, Z. Hu, H. Zhang, "A study of aero-engine control method based on deep reinforcement learning," *IEEE Access*, vol. 7, pp. 55285-55289, 2019.

[31] K. Miao, X. Wang, M. Zhu, S. Yang, X. Pei, Z. Jiang, "Transient controller design based on reinforcement learning for a turbofan engine with actuator dynamics," *Symmetry*, vol. 14, no. 4, p. 684, 2022.

[32] R.-R. Qian, Y. Feng, M. Jiang, L. Liu, "Design and realization of intelligent aero-engine DDPG controller," in *Journal of Physics: Conference Series*, vol. 2195, no. 1, p. 012056, 2022.

[33] B. Tao, L.-Y. Yang, D.-S. Wu, S.-L. Li, Z.-X. Huang, X.-S. Sun, "Deep reinforcement learning-based optimal control of variable cycle engine performance," in *International Conference on Advanced Robotics and Mechatronics (ICARM)*, pp. 1002-1005, 2022.

[34] Y. Zhu, M. Pan, W. Zhou, J. Huang, "Intelligent direct thrust control for multivariable turbofan engine based on

reinforcement and deep learning methods," *Aerospace Science and Technology*, vol. 131, p. 107972, 2022.

[35] J. Zhu, W. Tang, J. Dong, P. Li, "A virtual reinforcement learning method for aero-engine intelligent control," in *8th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pp. 138-143, 2023.

[36] J. Zhu, W. Tang, J. Dong, "Design of intelligent controller for aero-engine based on TD3 algorithm," *Information Technology and Control*, vol. 52, no. 4, pp. 1010-1024, 2023.

[37] W. Gao, M. Pan, W. Zhou, F. Lu, J.-Q. Huang, "Aero-engine modeling and control method with model-based deep reinforcement learning," *Aerospace*, vol. 10, no. 3, p. 209, 2023.

[38] X. Zhang, L. Zhonglin, J. Runmin, T. Zhang, "Deep reinforcement learning based active surge control for aeroengine compressors," *Chinese Journal of Aeronautics*, vol. 37, no. 7, pp. 418-438, 2024.

[39] Y. Zhu, M. Pan, W. Zhou, J. Huang, "Self-evolution direct thrust control for turbofan engine individuals based on reinforcement learning methods," *Aerospace Science and Technology*, vol. 144, p. 108734, 2024.

[40] H. Chen, S. Liu, W. Lyu, R. Dai, "Data-driven optimal thrust tracking control for a turbofan engine system with unknown dynamics based on adaptive dynamic programming," *Aerospace Systems*, pp. 1-10, 2025.

[41] O. Altuntas, "Designation of environmental impacts and damages of turbojet engine: A case study with GE-J85," *Atmosphere*, vol. 5, no. 2, pp. 307-323, 2014.

[42] A. Fawke, "Digital computer simulation of gas turbine dynamic behaviour," *PhD Thesis*, University of Bristol, 1970.

[43] J. Kurzke, "Preparing compressor maps for gas turbine performance modeling: Smooth C 9," *User's Manual*, 2002.

[44] J. Kurzke, "Preparing turbine maps for gas turbine performance modeling: Smooth T 9," *User's Manual*, 2003.

[45] S. Yarlagadda, "Performance analysis of J85 turbojet engine matching thrust with reduced inlet pressure to the compressor," *Master's Thesis*, University of Toledo, 2010.

[46] P. P. Walsh, P. Fletcher. Gas Turbine Performance. New Jersey: John Wiley & Sons, 2004.

[47] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, "Deterministic policy gradient algorithms," in *International Conference on Machine Learning*, pp. 387-395, 2014.

[48] R. Dong, J. Du, Y. Liu, A. A. Heidari, H. Chen, "An enhanced deep deterministic policy gradient algorithm for intelligent control of robotic arms," *Frontiers in Neuroinformatics*, vol. 17, p. 1096053, 2023.