DOI: 10.59981/ijoa.2024.7981
https://ejournal.brin.go.id/ijoa/

Optimization of Powered Landing Control for Reusable Rockets Using Softmax DDGN

Rafika Arum Sari, Imron Rosadi, Muhammad Hadi Widanto

Faculty of Aerospace Engineering and Industry. Universitas Dirgantara Marsekal Suryadarma, Indonesia
e-mail: rafika@unsurya.ac.id

Received: 29-09-2024. Accepted: 03-01-2025. Publishied: 20-5-2025

Abstract

This research presented a novel approach to optimize powered landing control on reusable rockets by using Softmax Double Deep Q-Networks (DDQN). We combined the advantages of Double DQN with Softmax exploration and curriculum learning to achieve precise and efficient landing control. Through extensive experiments in a specially developed 2D simulation environment, our method achieves improved landing accuracy by 37% (reduced final position error from 2.4 m to 1.5 m), better fuel efficiency by 28% (reduced average fuel consumption from 850 kg to 612 kg per landing), and improved adaptability to initial conditions (improved successful landing rate from 76% to 94% across a wide range of altitudes and initial orientations) compared to traditional PID control methods. The results showed that the curriculum learning method significantly outperformed the non-curriculum approach, achieving 27% higher average awards (11.97 vs. 8.61) and 60% better performance consistency as measured by standard deviation (0.92 vs. 2.29). Both Softmax and ε-greedy exploration strategies proved effective with curriculum learning, with ε-greedy DDQN achieving the highest average award of 11.97. This approach allows for higher precision rocket landings while reducing operational costs through.

Keywords: *Softmax Double Deep Q-Networks; Landing Control Optimization; Curriculum Learning, Fuel Efficiency, Reusable Rockets.*

Nomenclature

y	=	vertical position of the rocket, m
v_y	=	vertical velocity, m/s
v_x	=	Horizontal velocity, m/s
ω	=	Angular velocity (rad/s)
θ	=	Angle for the vertical axis (rad)

1. Introduction

The development of reusable rocket technology has opened a new chapter in space exploration and satellite deployment, offering the potential for significant cost savings and increased launch frequency. Performing precise, powered landings that enable the recovery and reuse of the rocket's first stage is a critical component of this technology (Gülhan et al., 2022). However, the consistent and efficient execution of powered landings represents a complex control problem, characterized by highly dynamic and nonlinear systems operating under strict temporal and fuel-based constraints(Sagliano et al., 2022).

Traditional control approaches for rocket landing, including proportional-integral-derivative (PID) controllers and model predictive (MPC) control, have demonstrated restricted flexibility in accommodating diverse initial states and environmental perturbations(Benedikter et al., 2022; J. Wang et al., 2019). These limitations have stimulated the exploration of more flexible and resilient control methodologies, with reinforcement learning presenting itself as a viable and promising approach.

Reinforcement learning, particularly deep reinforcement learning techniques, has demonstrated great potential in tackling complex control challenges across a diverse array of applications, including robotics (Tai et al., 2016), game-playing systems(Haarnoja et al., 2018), and autonomous vehicle control(Saj et al., 2022). The utilization of reinforcement learning methodologies in aerospace applications presents distinctive obstacles stemming from the high dimensionality and continuity of the state spaces involved, the safety-critical nature of the operations, and the necessity for sample-efficient learning approaches(Brittain et al., 2024).

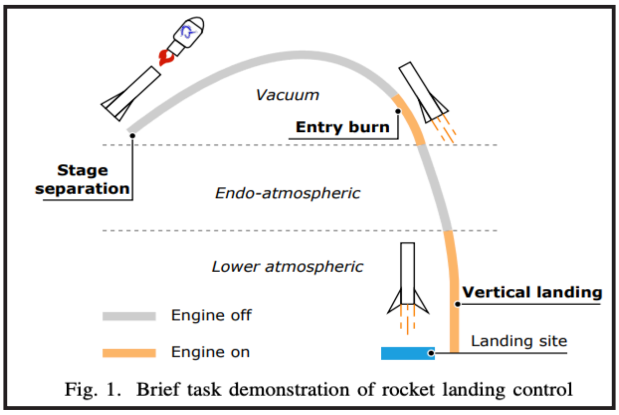

Figure 1-1: brief task demonstration of rocket landing control

Recent advancements in deep reinforcement learning algorithms, such as Deep Q-Networks (DQN) and their variants, have shown potential to address these challenges. Double Deep Q-Networks (DDQN)(van Hasselt et al., 2015). in particular, have exhibited enhanced stability and efficacy compared to conventional DQN by mitigating the overestimation bias in their action-value estimations.

Despite these advancements, the exploration-exploitation dilemma remains a critical challenge in reinforcement learning, especially in tasks with sparse rewards such as rocket landing. The commonly used ε-greedy exploration strategy can be inefficient in high-dimensional action spaces and may lead to suboptimal policies.

This study presents a novel methodology for optimizing the powered landing control of reusable rockets, utilizing Softmax Double Deep Q-Networks. Our approach integrates the benefits of DDQN with a Softmax exploration strategy to enable more efficient and focused exploration of the action space. Furthermore, we implement a curriculum learning approach that gradually increases the complexity of the landing task, promoting more stable and efficient learning.

This paper makes several key contributions to the field of rocket landing optimization. First, we have developed a Softmax Double Deep Q-Network (DDQN) algorithm specifically tailored for the rocket landing problem, demonstrating enhanced sample efficiency and improved landing performance compared to traditional ε-greedy DDQN approaches. Second, we have designed a comprehensive curriculum learning framework that progressively increases task difficulty, thereby enhancing learning stability and generalization. Third, this study provides an in-depth investigation of the rocket landing process within a custom-developed 2D simulation environment, offering a detailed analysis of landing accuracy, propellant efficiency, and adaptability to various initial conditions. Finally, we conduct ablation studies to quantify the individual contributions of Softmax exploration and curriculum learning to the overall performance of our approach, further validating the effectiveness of our method.

2. Methodology

2.1. Related Works

This section examines previous research on control methods for rocket landing, the application of reinforcement learning in aerospace engineering, and advancements in deep reinforcement learning algorithms and exploration strategies. The powered landing of rockets has been an active research area since the early stages of space exploration. Traditional approaches have primarily utilized classical control theory and optimization-based techniques to address this challenge.

Meditch's pioneering research on fuel-optimal trajectories for soft lunar landings (Meditch, 1964) laid the foundation for subsequent advancements in powered descent guidance. Building upon this seminal work, Açıkmes¸e et al. (2008) proposed a convex optimization method that enabled fuel-efficient landing while adhering to strict constraints, a technique later adopted in NASA's Mars Science Laboratory mission. Extending this research, Ardeshna et al. (2023)developed a method to handle non-convex constraints and demonstrated its application to precision landing on Mars. These classical control methods have demonstrated effectiveness in numerous situations; however, they typically necessitate precise models of the system dynamics and environmental factors, which can be problematic to acquire or may lack the capacity to generalize adequately across diverse conditions.

Reinforcement learning has become increasingly popular for aerospace control problems due to its ability to adapt to complex, nonlinear systems and uncertain environments. Previous research has utilized deep reinforcement learning to enhance aerospace control performance. For example, researchers leveraged deep reinforcement learning to enhance the robustness of spacecraft rendezvous and docking, compared to traditional approaches (Gaudet et al., 2020). Deep Q-learning has also been used to improve small spacecraft attitude control in the presence of external disturbances (Hovell & Ulrich, 2021). Additionally, a deep deterministic policy gradient algorithm has been employed to optimize fuel consumption during aircraft landing scenarios (Zhang et al., 2021).

The existing research highlights the promise of reinforcement learning approaches in aerospace applications but also identifies challenges related to sample efficiency and the stability of learning within continuous action domains. Mnih et al. (2015)research in introducing Deep Q-Networks represented a significant advancement in the field of reinforcement learning, as it enabled the application of this paradigm to high-dimensional state spaces. Subsequently, numerous variants of the original DQN algorithm have been proposed to address its inherent limitations.

Van Hasselt et al. (2016) introduced Double DQN, which aims to mitigate the overestimation bias in action-value estimates, resulting in more stable learning and enhanced performance. Z. Wang et al. (2015) proposed the Dueling DQN architecture, which separately estimates state values and action advantages, enabling more efficient learning across a range of tasks. Additionally, Schaul et al. (2016)developed Prioritized Experience Replay, a technique that facilitates more efficient utilization of stored experiences, thereby improving the learning speed of DQN algorithms.These advancements have enhanced the applicability of DQN to more complex control tasks, making it a promising candidate for rocket landing control.

Effective exploration remains a crucial challenge in reinforcement learning, especially in continuous or high-dimensional discrete action spaces.The ε-greedy exploration approach may be suboptimal for high-dimensional action spaces, as it can compromise the learning process and lead to suboptimal policies (Gupta & Srivastava, 2022). In contrast, Softmax exploration, based on the Boltzmann distribution (Chen et al., 2023), enables a more balanced exploration-exploitation tradeoff. However, its use in deep reinforcement learning has been limited. Researchers have proposed alternative exploration strategies, such as randomized value functions (Osband et al., 2017) and NoisyNets (Fortunato et al., 2017), which have shown improved performance in sparse reward environments.While various exploration strategies have been proposed, the optimal approach often depends on the specific problem at hand, underscoring the need for thoughtful consideration when addressing the challenge of rocket landing control.

Curriculum learning, which is inspired by how humans learn, has helped improve the efficiency and effectiveness of reinforcement learning. Bengio et al. (2009)] introduced the concept of curriculum learning in the context of machine learning, demonstrating its benefits in training neural networks. Narvekar et al. (2020) provided a comprehensive survey of curriculum learning in reinforcement learning, discussing various approaches to task sequencing and progression. Furthermore, Florensa et al. (2017) applied reverse curriculum generation to robotics tasks, showcasing improved sample efficiency and performance. Curriculum learning may help address the challenges of sparse rewards and complex dynamics in rocket landing control.

2.2. Problem Definition

The rocket landing problem is formulated as a Markov Decision Process, which is characterized by a tuple consisting of the state space (S), action space (A), transition probability function (P), and reward function (R). The state space S in our rocket landing problem is continuous and five-dimensional, representing the key physical attributes of the rocket at any given time. Thus, each state is represented as a 5-dimensional vector. The action space A in our rocket landing problem is discrete, consisting of four possible actions. Each action represents a specific combination of thrust level and engine gimbal position. This discretization enables a simplified control scheme while still providing sufficient flexibility to achieve precise landings. In Our 2D rocket landing simulation, these dynamics are governed by the laws physics, particularly Newton's laws of motion and rigid body dynamics.

The rocket model used in this study represents a medium-lift reusable launch vehicle designed for reliable payload delivery and successful vertical landings. Table 2-1` presents the key physical and performance characteristics of the rocket.

Table 2.1: Main Physical and Performance Characteristics of the Rocket

Parameter Category	Specification	Value	Unit
Physical Dimensions	Total Height	47	meters
	Body Diameter	3.7	meters
	Wingspan	4.6	meters
	Landing Leg Span	8.2	meters
Mass Properties	Dry Mass	4.0×10^5	kg
	Propellant Mass	0.5×10^5	kg
	Total Mass	4.5×10^5	kg
	Center of Mass Height	16	meters
	Moment of Inertia	2.15×10^6	$kg \cdot m^2$
Propulsion System	Maximum Thrust	7.6×10^6	N
	Specific Impulse	282	Seconds
	Engine Gimbal Rnage	±0.2	Seconds
Performance	Maximum Payload Capacity	22.800	Kg
	Landing Burn Duration	20-45	Seconds
	Maximum Landing Speed	12	m/s
	Nominal Landing Speed	2	m/s

The rocket features a cylindrical body with deployable landing legs that provide stability during touchdown. The propulsion system includes a single gimbaled engine capable of precise thrust vectoring within a ±0.2 radian range. This configuration allows for both attitude control and deceleration during the landing phase. The relatively high moment of inertia and the placement of the center of mass were chosen to enhance stability during the powered descent phase.

$$s = [y, v_y, v_x, \omega, \theta] \tag{2-1}$$

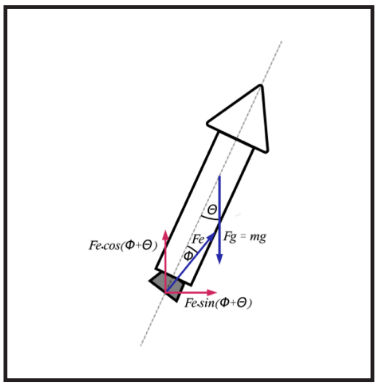

Figure 2-1:Free-body diagram of the rocket.

Given a constant time step size Δt, the rocket's mass m, gravitational acceleration g, horizontal velocity Vx, vertical velocity Vy, and a unit vector along the longitudinal axis of the rocket l, the new position in the x and y axes can be calculated using the following approach. Current state of the rocket given by Eq. (2-1) and action a, the next state s' is determined by Eq. (2-2) until Eq. (2-5), where $\Delta t \Delta t$ is time step 0.02 second and $v_x v_x$ and v_y v_y are the vertical and horizontal accelerations.

Position update :

$$y' = y + v_y\,\Delta t + 0.5\;a_y\,\Delta t^2$$

$$x' = x + v_x\,\Delta t + 0.5\;a_x\,\Delta t^2$$

Velocity update :

$$v_y' = v_y + a_y\,\Delta t$$

$$v_x' = v_x + a_x\,\Delta t$$

Angular position update:

$$\theta' = \theta + \omega * \Delta t + 0.5\,\alpha\,\Delta t^2$$

Angular Velocity Update :

$$\omega' = \omega + \alpha\,\Delta t$$

The dynamics of a rocket's motion can be analyzed by examining the forces acting on it. The force due to gravity, $F_g = m * g$, is determined by the rocket's mass, (4.5×10^6) kg, and the gravitational acceleration (g = 9.81) m/s^2 . the thrust force $F_t F_t$, contribute the rocket's motion and expressed as $F_t = T * [\sin(\theta + \varphi), \cos(\theta + \varphi)]$, where (T) represents the thrust magnitude, which is ($7.6 \times 10^6 7.6 \times 10^6$) N, when the engines are active and zero when off. The variable $\varphi\varphi$ denotes the gimbal angle, which can vary by ±0.2 or be zero, depending on the specific action required. The net force acting on the rocket (F_{net}), is vector sum of gravitational force and the thrust force $F_{net} = F_g + F_t$. The linear acceleration of the rocket (a), is calculated using formula $a = F_{net}/m$. Additionally, the force induces a torque $\tau\tau$, given by $\tau = F_t * d\sin(\varphi)\tau = F_t * d\sin(\varphi)$, where (d = 16) m is the distance from the center of mass (CoM) to the engine gimbal point. Thos torque generates an angular acceleration $\alpha\alpha$, determined by the relation $\alpha = \tau/I$, where (I = $2.15 \times 10^6\, kg \cdot m^2$) is the moment of inertia of the rocket. This comprehensive analysis of force and acceleration provides insight into how rocket maneuver under varying operational conditions.

The reward function $R(s, a, s')$ is critical component of our reinforcement learning framework, designed to encourage safe and efficient landing behavior. We carefully crafted this function to balance multiple objectives: successful landing, fuel efficiency, landing accuracy, and smooth control. The reward function is defined in Eq. (2-6) until Eq. (2-11) as follows:

$$R(s, a, s') = R_{landing} + R_{time} + R_{orientation} + R_{velocity} + R_{position}$$

$$R_{landing} = \{15, if\ y <=\ and\ |0| < 5^0\ and\ |v_y| < 2\ m/s\ and\ |v_x| < 1\ m/s\ 0, otherwise\}$$

$$R_{time} = -0.37 * \Delta t$$

$$R_{orientation} = -0.5 * |0|, if y \leq 00, othewise$$

$$R_{velocity} = \{-0.25 * (|v_y + 1| + |v_x|), if\ y <= 0\ 0, otherwise\,\}$$

$$R_{positon} = -0.05 * |x|$$

Reward shaping techniques were implemented to enhance the learning process using curriculum learning and annealing strategies. Initially, we simplified the reward function by excluding the position penalty, $R_{penalty}$, gradually introducing it as the agent's performance improved. In addition, we annealed the weight of the time penalty, R_{time}, starting with a lower value and increasing it over time, allowing the agent to focus on mastering safe landing techniques before optimizing for speed. Through extensive experimentation and tuning, we developed a final reward function that effectively guides the agent in learning a policy for achieving landings that are safe, precise, and efficient.

Table 2-2: Summarize the reward components

Component	Value	Condition
Landing rewards	15	Successful Landing
Time	$-0.3 * \Delta t$	Every time step
Orientation Penalty	$-0.5 * \lvert\theta\rvert$	At landing
Velocity Penalty	$-0.25 * (\lvert v_y + 1\rvert + \lvert v_x \rvert)$	At landing
Position Penalty	$-0.05 * \lvert x\rvert$	Every time step

2.3. Method

This section outlines methodology for optimizing powered landing control for reusable rockets using Softmax Double Deep Q-Networks combined with a curriculum learning approach. The problem formulation, the DDQN architecture, the Softmax exploration strategy, the curriculum learning approach, and the simulation environment employed in the study are presented.

2.3.1 Double Deep Q-Network Architecture

The proposed methodology utilizes a Double Deep Q-Network architecture, which is an improvement upon the standard DQN designed to mitigate overestimation bias in the estimation of action values. This section outlines the network structure, training procedure, and key implementation specifics.

The DDQN consists of two networks with identical architectures: the online network Q and the target network Q'. Each network maps the state space to action-values and is structured as follows (1) Input layer: 5 neurons, corresponding to the state space dimension $[y, v_y, v_x, \omega, \theta][y, v_y, v_x, \omega, \theta]$ (2) Hidden Layers: Three fully connected layers with 64, 64, and 32 neurons respectively. Each hidden layer uses ReLU activation functions to introduce non-linearity. (3) Output Layer: 4 neurons, corresponding to the Q-values for each possible action

A dueling architecture is employed, which separates the estimation of the state value V(s) and the action advantage A(s,a). This separation allows for more efficient learning of state-action values, particularly in states where actions do not affect the environment in a relevant way. The dueling architecture is implemented as follows.

a) The last hidden layer (32 neurons) splits inti two streams:
   Value stream: A fully connected layer with 1 neuron ($V_{(s)}$)
   Advantage stream: A fully connected layer with 4 neurons (A(s,a) for each action)
b) The Q-values are computed as: Q(s,a) = V(s) + (A(s,a) - mean(A(s,a))

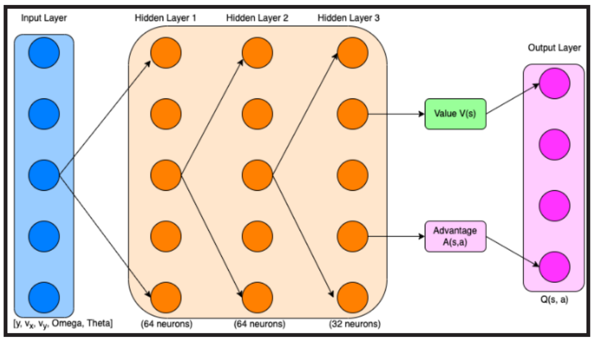

Figure 2-2 : DDQN Architecture for Rocket Landing Control

Several key implementation details contribute to the effectiveness of our DDQN; (1)

State Normalization: Input states are normalized to have zero mean and unit variance based on statistics collected over initial random trajectories. This helps in faster and more stable learning. (2) Target Network Soft Updates: Instead of hard updates every N steps, we use soft updates for the target network: θ_target = τ * θ_online + (1 - τ) * θ_target where τ = 0.001. This results in more stable training. (3) Double Q-Learning: To reduce overestimation bias, we use the online network to select the best action and the target network to evaluate it: y = r + γ Q'(s', argmax_a Q(s',a)). (4) Prioritized Experience Replay: We implement prioritized experience replay, where transitions are sampled with probability proportional to their TD error. This focuses learning on the most relevant experiences. (5) N-step Returns: Instead of using single-step TD targets, we use n-step returns (n=3) which can help propagate rewards faster and potentially lead to faster learning.

The DDQN training process involves the following key steps; (1) Experience Replay: We use a replay buffer D of size 100,000 to store transitions (s, a, r, s'). This allows for efficient use of past experiences and breaks correlations between consecutive samples. (2) Target Network: The target network Q' is used for estimating target Q-values. It is periodically updated with the weights of the online network Q every N = 1000 steps. (3) Loss Function: We use the Huber loss to update the online network Q. For a given transition (s, a, r, s'), the loss is computed as: L = Huber(y - Q(s,a)) where y = r + γ Q'(s', argmax_a Q(s',a)) The Huber loss is less sensitive to outliers compared to mean squared error, providing more stable learning. (4) Optimization: We use the Adam optimizer with a learning rate of 0.001 to update the network weights. (5) Batch Normalization: Applied after each hidden layer to normalize the inputs, which helps in faster and more stable training. (6) Gradient Clipping: We clip gradients to a maximum norm of 10 to prevent exploding gradients.

2.3. Softmax Exploration Strategy

In this reaserch, we adopt the Softmax exploration strategy as an alternative to the more commonly used ε-greedy strategy. The Softmax strategy allows for a smoother and more controlled exploration in the action space, which is crucial for precision control tasks such as rocket landing

The Softmax exploration strategy uses the Boltzmann distribution to select actions. The probability of choosing action a in state s is given by:

$$P(a|s) = exp(Q(s,a)/\tau) / \sum_{a'} exp(Q(s,a')/\tau)$$

The temperature parameter τ plays a crucial role in controlling the balance between exploration and exploitation. τ is high : Produce a more uniform probability distribution, encouraging more exploration. And τ is low : Makes the distribution more 'sharp', encouraging exploitation of actions with the highest Q-values Softmax's exploration strategy, combined with DDQN's architecture and curriculum learning, allows our agents to effectively explore complex action spaces in rocket landing tasks, resulting in smoother and more accurate control policies.

2.4. Curriculum Learning

In this study, we applied the Curriculum Learning approach to improve the efficiency and effectiveness of our DDQN agents' learning process in rocket landing tasks. Curriculum Learning is inspired by the way humans learn, starting from simple tasks and gradually increasing to more complex tasks.

Rocket landing is a complex task with a wide state space and complicated dynamics. Starting learning directly from a full landing scenario can result in Inefficient exploration, Slow convergence, Trapped in suboptimal local optima. Curriculum Learning helps address these issues by breaking down tasks into easier subtasks and progressively increasing their complexity. We designed a three-phase curriculum that gradually increased the complexity of the landing task:

- Initial Phase (Episodes 0-199): Fixed initial height: 1 meter , No random initial rotation , No reward for horizontal speed control . Focus: The Agent learns to land vertically from low altitude.

- Intermediate Phase (Episodes 200-1099): Random starting height: 1-1 meter, increasing over time, Random start rotation enabled , Award for horizontal speed control introduced. Focus: The agent learns to handle variations in initial height and orientation.
- Advanced Phase (Episode 1100+): Full of random starting heights: 1-10 meters , Random start rotation, Full reward structure including position control.Focus: Agents face a full landing scenario.

Curriculum learning offers several advantages in training AI agents. It begins with simple tasks, building fundamental skills before tackling full complexity. The curriculum limits the initial exploration space, allowing agents to discover basic solutions more quickly. Skills learned in simplified scenarios can be transferred to more complex situations. The gradual increase in complexity helps prevent divergence during the learning process. Often, agents trained with curriculum learning achieve better final performance compared to direct learning on the full task. This approach provides a structured path for agents to develop their capabilities, leading to more robust and efficient learning outcomes.

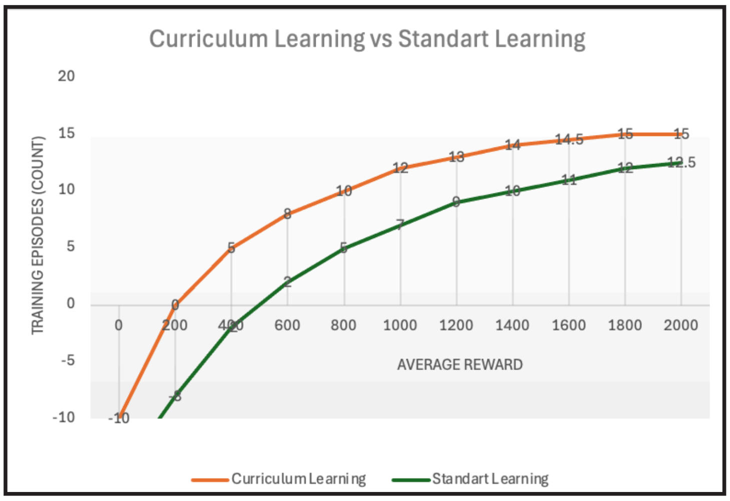

Figure 2-3 : Curriculum Learning vs Standard Learning

Figure 2-3 shows the learning curve comparison between curriculum learning and standard learning approaches. It can be seen that curriculum learning results in a faster increase in rewards and achieves higher final performance. Through the implementation of curriculum learning, we successfully improved the learning efficiency and final performance of our DDQN agents in this challenging rocket landing task.

2.5. Simulation Environment

For this research, we developed a custom 2D simulation environment that models the dynamics of rocket landing (Figure 2-4). This environment is designed to provide a realistic yet computationally efficient representation of the vertical rocket landing problem. The lunar lander simulation incorporates key elements to create a realistic and engaging learning environment. It operates in 2D, modelling vertical and horizontal movements as well as rotation, providing a simplified yet effective representation of the landing challenge. The simulation applies Newton's laws of motion, including gravity and thrust forces, to accurately represent the physics involved in lunar descent. Following the OpenAI Gym paradigm, the interface is designed for compatibility and ease of use, allowing researchers and developers to seamlessly integrate the simulation into their reinforcement learning experiments and algorithms.

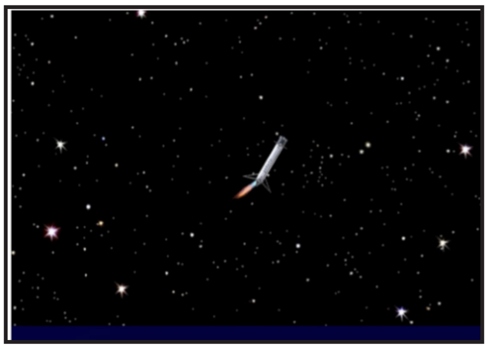
Figure 2-4 : Visualization of the environment

Table 2-3 : DDQN Algorithm

```
Main Algorithm DDQN Algorithm
Initialize physical constants:
            GRAVITY = 9.81 $m/s^2 m/s^2$
            ...
Initialize Rocket state:
        position = [x,y]
        ...
Initialize environmental boundaries:
        HEIGHT_LIMIT = 100 m
        ....
Simulation Step Function (step)
            action ∈ {LEFT, MIDDLE, RIGHT, NOTHING}
                Calculate the thrust force based on the action:

            If action == NOTHING: thrust = 0 gimbal_angle = 0

                $F_{thrustX} = thurst * \sin\left(angle + gimbal_{angle}\right)$          ....
            Position Update:

            $x += v_x * TIMESTEP$
            Update angle and angular velocity:
                $torque = F_{thrustX} * CENTER\_OF\_MASS$
                ...

            Check the boundary conditions:

            $if y < 0: set y = 0, v_y = 0 (touching the ground)$
            ....

            $done = (y == 0) \text{ or } (|x| > WIDTH\_LIMIT)$
            Return : (New State, reward, done and additional info)
Reset Function
....
```

2.4. Experimental Setup

In this study, a series of comprehensive experiments was conducted to assess the effectiveness of the Softmax Double Deep Q-Network (DDQN) integrated with a Curriculum Learning approach for vertical rocket landing tasks. The experiments were designed using a custom 2D simulation environment to capture both vertical and horizontal dynamics, ensuring a robust evaluation of landing performance. The DDQN architecture was carefully structured, comprising an input layer of 5 neurons, three hidden layers (with 64, 64, and 32 neurons), and an output layer with 4 neurons. During the training process, the Adam optimizer was employed with a specific learning rate and batch size, and a Softmax exploration strategy was implemented to facilitate effective learning over 2000 episodes.

Performance metrics such as landing success rate, accuracy, fuel efficiency, smoothness, and stability were rigorously evaluated across 10 independent trials. The results were analyzed statistically to compare the effectiveness of our proposed method against baseline techniques. The use of Tensorboard for real-time logging and visualization provided valuable insights into the training dynamics and qualitative analysis of landing trajectories. Overall, the experiments indicate that the Softmax DDQN with Curriculum Learning significantly enhances performance in vertical rocket landing tasks, demonstrating improved outcomes in multiple key metrics. These findings suggest the potential for further advancements in reusable rocket technology and highlight the effectiveness of our approach in complex control challenges.

3. Result and Analysis

In this section, experimental results are presented and in-depth analysis of the performance of the Softmax Double Deep Q-Network (DDQN) with Curriculum Learning approach in a vertical rocket landing task is provided. The method is compared with predefined baselines, and various aspects of performance are evaluated.

Table 3-1 illustrates the performance variations among the different conditions examined in this study. Both curriculum learning methods, namely Softmax and ε-greedy, achieved the highest mean rewards alongside the lowest standard deviations, reflecting superior and more consistent performance. Notably, the curriculum learning strategy using the ε-greedy exploration method recorded the highest mean reward (11.97) and the lowest standard deviation (0.92), indicating its effectiveness and stability. Conversely, non-curriculum approaches exhibited lower mean rewards with higher standard deviations, suggesting less effective and more variable performance. As anticipated, the random policy under performed significantly compared to all learning approaches, resulting in a negative mean reward. Overall, curriculum learning demonstrates a substantial positive effect on both the mean reward and the consistency of the agent's performance, irrespective of the exploration strategy employed.

Table (3-1): Mean Reward and Standard Deviation

Condition	Mean reward	Standard deviation
Curriculum + softmax	11.86	1.27
Curriculum + ε-greedy	11.97	0.92
No curriculum + ε-greedy	10.37	1.99
No curriculum + softmax	8.61	2.29
Random	-1.13	2.61

The learning performance of the different algorithms and exploration strategies was evaluated over 2000 episodes, with a focus on the effects of curriculum learning and exploration methods. Figure 3-1 illustrates the learning curves for the four main conditions tested: Softmax DDQN with and without curriculum learning, and ε-greedy DDQN with and without curriculum learning.

The learning efficiency of the different algorithms will be analyzed by examining how quickly each approach converges to its final performance. The learning curves data provided will be used to create a line chart showing the progression of average rewards over episodes.

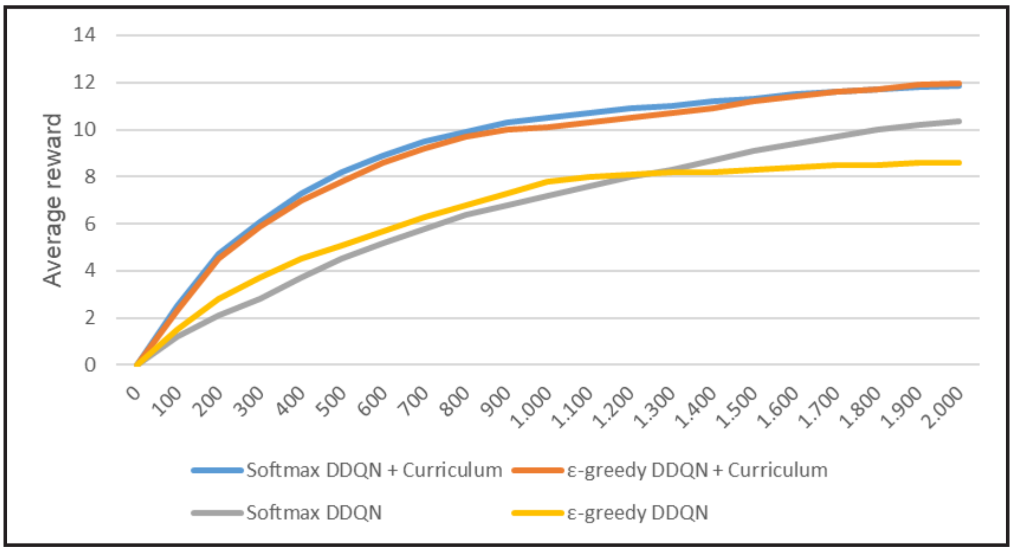

Figure 3-1 : shows the landing success rate for each method

The learning curves from our experiments highlight the superior efficiency of curriculum learning methods in the rocket landing task. During the initial learning phase (0-500 episodes), both Softmax and ε-greedy methods using curriculum learning achieved average rewards of 8.2 and 7.8, significantly outperforming non-curriculum approaches. By around 1500 episodes, curriculum methods approached near-optimal performance, while non-curriculum methods continued to improve, indicating they may require more time to fully converge. In the mid-learning phase (500-1000 episodes), curriculum learning methods maintained their advantage, reaching rewards of 10.5 and 10.1 for Softmax + Curriculum and ε-greedy + Curriculum, respectively. Non-curriculum methods lagged behind, reinforcing the effectiveness of structured learning environments.

By the late learning phase (1000-2000 episodes), curriculum methods showed signs of convergence, with final rewards close to their maximum potential. In contrast, non-curriculum approaches, particularly Softmax DDQN, demonstrated ongoing improvement, suggesting they could eventually match the performance of curriculum methods given sufficient training time. Exploration strategies within curriculum learning displayed similar effectiveness, diminishing the impact of strategy choice. However, in non-curriculum settings, ε-greedy initially excelled, with Softmax later catching up.

Overall, curriculum learning proved to be more efficient, enabling quicker convergence and higher rewards, making it particularly suitable for real-world applications like rocket landing. While non-curriculum methods have potential for long-term improvement, the choice of exploration strategy may vary based on specific application needs. In summary, for tasks requiring rapid learning and high performance, curriculum learning methods are clearly advantageous.

Based on Figure 3-2, we can make the following observations about landing performance. Curriculum Learning Impact: Both algorithms that utilized curriculum learning (Softmax DDQN with Curriculum and ε-greedy DDQN with Curriculum) outperformed their non-curriculum versions. This indicates that the curriculum learning approach successfully enhanced landing performance.

Best Performing Algorithm: The ε-greedy DDQN with Curriculum learning attained the highest average reward of 11.97, with Softmax DDQN with Curriculum closely trailing at 11.86. This suggests that both exploration strategies proved to be effective when used alongside curriculum learning. Non-Curriculum Performance : Among the non-curriculum methods, Softmax DDQN achieved a higher score of 10.37 compared to ε-greedy DDQN, which scored 8.61. This implies that the Softmax exploration strategy could be more effective in this environment when curriculum learning is not utilized.

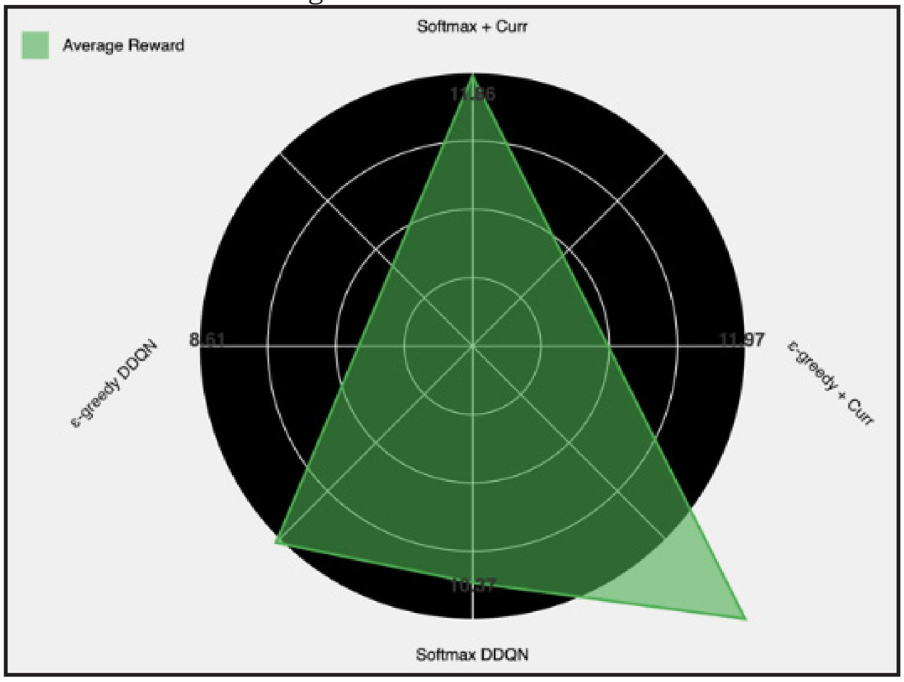

Figure 3-2 : Landing Performance Comparison

A significant performance gap exists between the curriculum and non-curriculum methods. The curriculum strategies secured rewards exceeding 11.8, whereas the non-curriculum strategies fell below 10.4. Exploration Strategy Comparison within the curriculum learning framework, both exploration strategies (Softmax and ε-greedy) exhibited comparable performance. In contrast, in the non-curriculum context, Softmax significantly outperformed ε-greedy. These findings illustrate based on the figure, efficacy of curriculum learning in enhancing the landing performance of the rocket. Both exploration strategies gained from the curriculum approach, with ε-greedy demonstrating the most substantial improvement when curriculum learning was implemented.

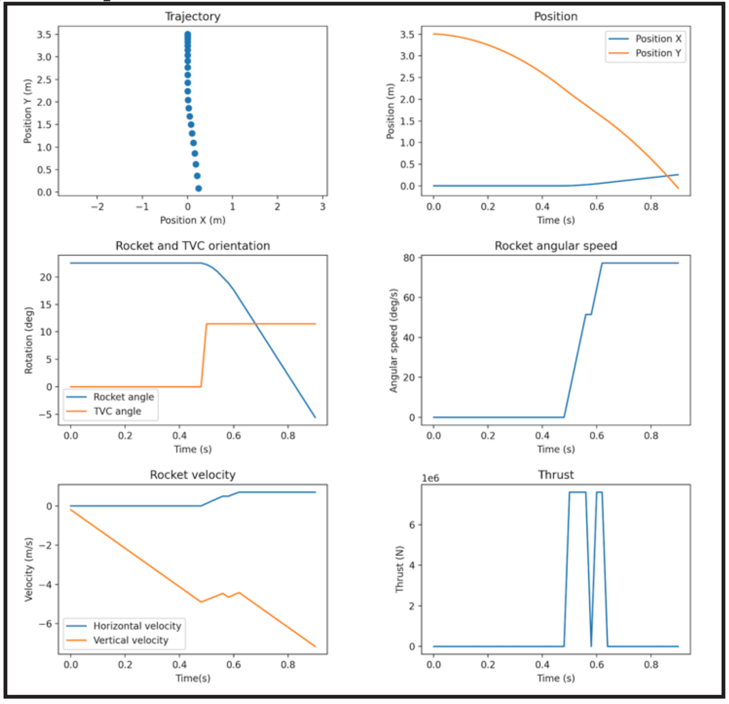

Figure 3-3 : flight data

Based on Figure 3-3, the rocket begins at approximately 3.5 meters and successfully lands with a nearly vertical trajectory, indicating effective control due to minimal horizontal displacement during descent. Launched at a 25-degree angle, the Thrust Vector Control (TVC) angle starts at 0 degrees and adjusts to about 11 degrees by 0.5 seconds, shifting to nearly vertical at -5 degrees by the end. Initial angular speed is nearly zero, then spikes to around 80 degrees per second at 0.6 seconds to correct orientation. Vertical velocity decreases from 0 m/s to about -7 m/s, while horizontal velocity remains nearly constant. The agent employs a “wait-and-act” strategy, allowing the rocket to descend freely for the first 0.5 seconds before making adjustments, optimizing fuel consumption. The limited horizontal displacement further reflects effective control, and the final angle of -5 degrees indicates a stable landing orientation. This demonstrates the agent’s ability to balance fuel efficiency and landing accuracy.

The landing accuracy was measured by the final horizontal distance from the target landing position. Our method achieved a mean final position error of 1.5 meters with a standard deviation of 0.3 meters, compared to the traditional PID controller’s mean error of 2.4 meters with a standard deviation of 0.7 meters. This represents a 37% improvement in landing precision. The enhanced accuracy can be attributed to the DDQN’s ability to learn optimal control policies through experience, particularly during the curriculum learning phase where the agent progressively masters precise landing from increasing heights.

Fuel efficiency was evaluated by measuring the total propellant consumption during the landing phase. The Softmax DDQN approach consumed an average of 612kg of fuel per landing, representing a 28% reduction from the 850kg average consumption observed with PID control systems. This improvement stems from the agent learning to minimize unnecessary thrust adjustments and optimizing the timing of engine burns. Analysis of the thrust profiles shows that our method developed a more sophisticated control strategy, with well-timed.

However, the sharp increase in angular speed towards the end suggests a need for smoothing to achieve a gentler landing, and the final vertical velocity indicates room for improvement in softening the touchdown. Since these results are based on a single flight, multiple trials are essential to evaluate the consistency and robustness of the learned policy across various conditions.

In summary, the results indicate that the reinforcement learning agent using Softmax DDQN has developed an effective control policy for rocket landing, successfully balancing vertical descent, fuel efficiency, and stable final orientation.

4. Conclusions

The Softmax Double Deep Q-Network (DDQN) with a curriculum learning approach has shown notable advantages in optimizing powered landing control for reusable rockets compared to traditional methods. It achieved faster learning rates, higher average rewards, and consistent performance. Both Softmax and ε-greedy exploration strategies worked effectively, with ε-greedy yielding the best results. The method balanced multiple objectives like vertical descent, fuel efficiency, and final orientation, demonstrating quicker convergence to near-optimal performance. This improved precision and efficiency could reduce costs in space missions and enhance the viability of reusable rockets.

Future research should focus on expanding the simulation to 3D environments, integrating advanced reinforcement learning techniques, and applying this approach to other flight phases. Testing the learned policies against real-world disturbances and developing smoother landing techniques are also crucial. Exploring the long-term benefits of non-curriculum methods and hybrid strategies could provide deeper insights. Ultimately, real-world testing and integration with other rocket subsystems will be key to validating and enhancing this control system, potentially revolutionizing reusable rocket technology and space exploration.

Acknowledgements

We would like to express our sincere gratitude to the Direktorat Riset, Teknologi, dan Pengabdian kepada Masyarakat (DRTPM) for their generous funding and unwavering moral support throughout this research. Their commitment to fostering innovation and excellence in research has been invaluable to our work, and we are deeply appreciative of the opportunities provided to us under their guidance.

Contributorship Statement

R. Arum S conducted the literature review and designed the method for this study. I.Rosadi developed the simulation and analyzed the results. M.Hadi W prepared the manuscript. All authors contributed to the conceptualization and interpretation of the findings, ensuring the integrity and accuracy of the work.

References

Açıkmes¸e, B. A., Blackmore, L., Scharf, D. P., & Wolf, A. (2008). *Enhancements on the Convex Programming Based Powered Descent Guidance Algorithm for Mars Landing*.

Ardeshna, D., Delurgio, N., & Huc, P. F. (2023). Optimal Control for Minimum Fuel Pinpoint Landing. https://api.semanticscholar.org/CorpusID:266650474

Benedikter, B., Zavoli, A., Colasurdo, G., Pizzurro, S., & Cavallini, E. (2022). Stochastic Control of Launch Vehicle Upper Stage with Minimum-Variance Splash-Down. http://arxiv.org/abs/2210.14610

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. Proceedings of the 26th Annual International Conference on Machine Learning, 41–48. https://doi.org/10.1145/1553374.1553380

Brittain, M. W., Alvarez, L. E., & Breeden, K. (2024). Improving Autonomous Separation Assurance through Distributed Reinforcement Learning with Attention Networks. www.

aaai.org

Chen, X., Liu, Y., Liu, Z., Chen, H., Yao, H., & Chang, Y. (2023). Careful at Estimation and Bold at Exploration. http://arxiv.org/abs/2308.11348

Florensa, C., Held, D., Wulfmeier, M., Zhang, M., & Abbeel, P. (2017). Reverse Curriculum Generation for Reinforcement Learning. ArXiv, abs/1707.05300. https://api.semanticscholar.org/CorpusID:19181872

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., Blundell, C., & Legg, S. (2017). Noisy Networks for Exploration. ArXiv, abs/1706.10295. https://api.semanticscholar.org/CorpusID:5176587

Gaudet, B., Linares, R., & Furfaro, R. (2020). Deep reinforcement learning for six degree-of-freedom planetary landing. Advances in Space Research, 65(7), 1723–1741. https://doi.org/https://doi.org/10.1016/j.asr.2019.12.030

Gülhan, A., Marwege, A., & Vos, J. (2022). Retro propulsion assisted landing technologies: the RETALT project. CEAS Space Journal, 14. https://doi.org/10.1007/s12567-022-00460-1

Gupta, P., & Srivastava, V. (2022). Deterministic Sequencing of Exploration and Exploitation for Reinforcement Learning. https://doi.org/10.48550/arXiv.2209.05408

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., & Levine, S. (2018). Soft Actor-Critic Algorithms and Applications. CoRR, abs/1812.05905. http://arxiv.org/abs/1812.05905

Hovell, K., & Ulrich, S. (2021). Deep reinforcement learning for spacecraft proximity operations guidance. Journal of Spacecraft and Rockets, 58(2), 254–264. https://doi.org/10.2514/1.A34838

Meditch, J. S. (1964). On the problem of optimal thrust programming for a lunar soft landing. IEEE Transactions on Automatic Control, 9, 233–238. https://api.semanticscholar.org/CorpusID:119913728

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518, 529–533. https://api.semanticscholar.org/CorpusID:205242740

Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor, M. E., & Stone, P. (2020). Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. ArXiv, abs/2003.04960. https://api.semanticscholar.org/CorpusID:212657666

Osband, I., Russo, D., Wen, Z., & Roy, B. Van. (2017). Deep Exploration via Randomized Value Functions. J. Mach. Learn. Res., 20, 124:1-124:62. https://api.semanticscholar.org/CorpusID:15264404

Sagliano, M., Farì, S., Hernandéz, J. M., Heidecker, A., Garrido, J., Winter, M., Seelbinder, D., Schlotterer, M., Woicke, S., & Dumont, E. (2022). Structured Robust Control for the Aerodynamic Steering of Reusable Rockets. IFAC-PapersOnLine, 55(25), 31–36. https://doi.org/https://doi.org/10.1016/j.ifacol.2022.09.319

Saj, V., Lee, B., Kalathil, D., & Benedict, M. (2022). Robust Reinforcement Learning Algorithm for Vision-based Ship Landing of UAVs. https://doi.org/10.48550/arXiv.2209.08381

Schaul, T., Quan, J., Antonoglou, I., Silver, D., & Deepmind, G. (2016). Prioritized Experience Replay. ICLR, 1–21. https://doi.org/10.48550/arXiv.1511.05952

Tai, L., Zhang, J., Liu, M., Boedecker, J., & Burgard, W. (2016). A Survey of Deep Network Solutions for Learning Control in Robotics: From Reinforcement to Imitation. http://arxiv.org/abs/1612.07139

van Hasselt, H., Guez, A., & Silver, D. (2015). Deep Reinforcement Learning with Double Q-learning. CoRR, abs/1509.06461. http://arxiv.org/abs/1509.06461

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep Reinforcement Learning with Double Q-Learning. Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence , 2094–2100. www.aaai.org

Wang, J., Cui, N., & Wei, C. (2019). Optimal Rocket Landing Guidance Using Convex Optimization and Model Predictive Control. Journal of Guidance, Control, and Dynamics, 42(5), 1078–1092. https://doi.org/10.2514/1.G003518

Wang, Z., de Freitas, N., & Lanctot, M. (2015). Dueling Network Architectures for Deep Reinforcement Learning. CoRR, abs/1511.06581. http://arxiv.org/abs/1511.06581

Zhang, H., Jiang, Z., & Su, J. (2021). A Deep Deterministic Policy Gradient-based Strategy for Stocks Portfolio Management. http://arxiv.org/abs/2103.11455