Reinforcement Learning Final Project: Solving the LunarLander Environment

Clément Leprêtre Cyrielle Théobald

CentraleSupélec

Objective

In this project, we trained an autonomous agent to safely land a spacecraft on the Moon in the OpenAI Gym Lunar Lander environment using Reinforcement Learning (RL). We first solved the discrete version with a **Deep Q-Network (DQN)**, then tackled the continuous action space using **Proximal Policy Optimization (PPO)**, both achieving the performance target.

LunarLander Game Overview

Environment Dynamics

The LunarLander operates in a **2D physics-based** simulation, affected by **gravity**, **inertia**, and terrain interactions, with control via four discrete actions.

Action Code	Description
0	Do nothing.
1	Fire left engine.
2	Fire main engine.
3	Fire right engine.

Table 1. Lander Control Actions

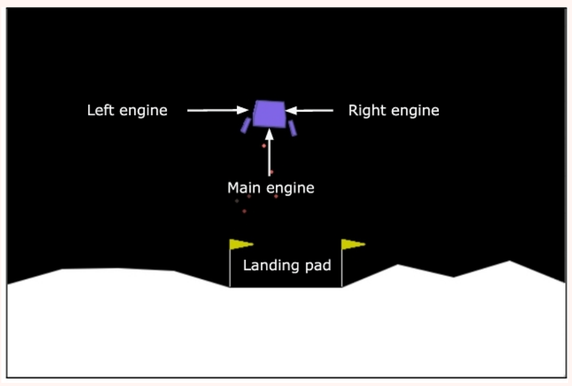

Figure 1. Lunar Lander environment in OpenAI Gym.

The lander's state is defined by eight continuous variables (position, velocity, orientation, leg contact).

Variable	Description
X position	Horizontal position of the lander.
Y position	Vertical position of the lander.
X velocity	Horizontal velocity component.
Y velocity	Vertical velocity component.
Angle	Orientation of the lander relative to the ground.
Angular velocity	Rate of change of the lander's angle.
Left leg contact with ground	Boolean indicating if the left leg is touching the ground.
Right leg contact with ground	Boolean indicating if the right leg is touching the ground.

Table 2. Lander State Variables

Reward Structure

The reward system encourages successful landing and fuel efficiency, with **200+ points** indicating success.

Solution in a discrete environment

Deep Q-Network (DQN)

Deep Q-Network (DQN) approximates the Q-value function with a neural network, using:

- **Experience Replay**: Breaks state correlation for stable learning.
- **Target Network**: Stabilizes training with consistent targets.
- **Epsilon-Greedy Exploration**: Balances exploration/exploitation with probability ϵ.

Hyperparameter	Value
Number of Episodes	600
Discount Factor (γ)	0.7
Learning Rate (α)	0.001
Hidden Layer Size	50
Gradient Clipping (Max Norm)	0.5
Replay Buffer Size	50000
Batch Size	54

Table 3. Hyperparameters used for training the Lunar Lander agent

Training Performance Analysis

Our optimized neural network has two hidden layers with 50 neurons each, sufficient for high learning capacity. Results in Figure 2 show the first implementation barely reached the 200-point target.

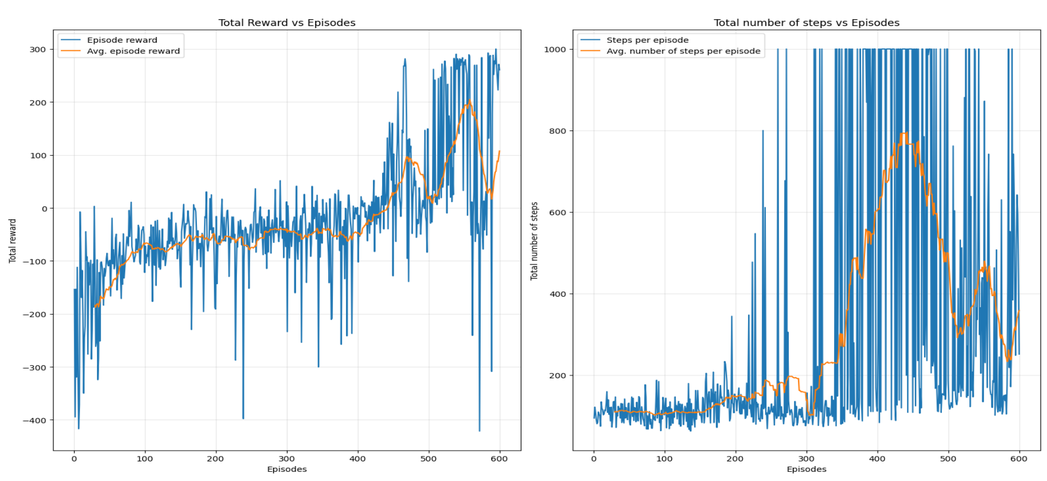

Figure 2. Standard training of the DQN

Solution in a continuous environment

Continuous Space

The transition from **LunarLander** to **LunarLanderContinuous** marks a fundamental shift in how the agent interacts with the environment. While the standard LunarLander employs a discrete action space with four possible actions (do nothing, fire left engine, fire main engine, fire right engine), LunarLanderContinuous instead utilizes a **continuous action space**. The following table summarize the changes between the two environments.

Feature	LunarLander	LunarLanderContinuous
Action Space	Discrete (4 actions)	Continuous (2 values)
Available Actions	0: Do nothing 1: Fire left engine 2: Fire main engine 3: Fire right engine	Throttle: [-1, 1] Steering: [-1, 1]
Control Granularity	Fixed set of actions	Smooth control over thrust and rotation

Table 4. Comparison of LunarLander and LunarLanderContinuous environments.

This change requires adapting the learning algorithm, as DQN is inherently designed for discrete action spaces, whereas PPO can naturally handle continuous control problems through stochastic policy optimization.

Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm belonging to the family of *policy gradient methods*, which iteratively updates policies using gradient descent to improve performance. The key idea behind PPO is to constrain the extent to which the policy can change in a single update, thereby ensuring stable learning. This is achieved by optimizing a surrogate objective function that incorporates a clipped probability ratio, preventing overly large policy updates.

At each iteration, PPO collects a batch of experiences using the current policy, then updates the policy by maximizing the clipped surrogate objective function. The algorithm follows an actor-critic framework, where the actor is responsible for updating the policy, while the critic estimates the value function to guide policy improvement.

Training Performance Analysis

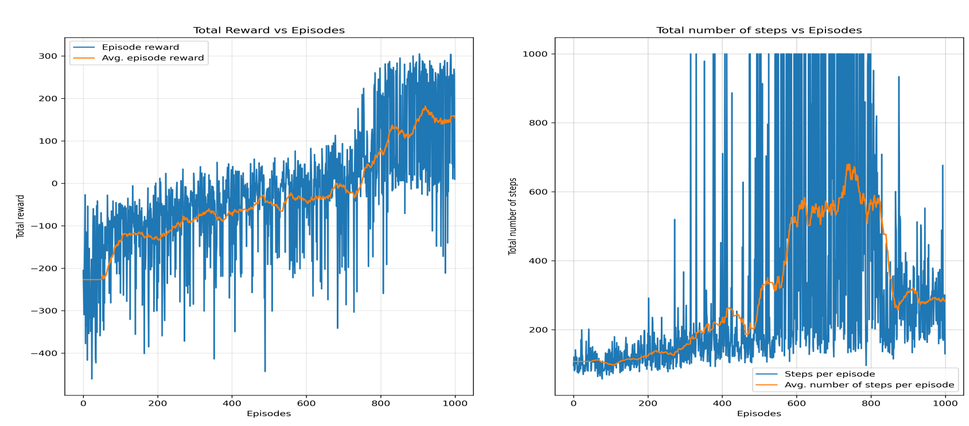

Figure 3. PPO basic training

Training performance of Proximal Policy Optimization can be evaluated by looking at both the episodic reward and number of steps. They reflect the ability of the algorithm to balance exploration and exploitation.

- Total reward per episode: The total reward increases over time, since the agent is learning to land the spacecraft more effectively.
- Steps per episode: After some point (step 700), the number of steps per episode decreases, showing the agent's ability to optimize its actions to achieve the goal in fewer steps.

Impact of a discrete vs continuous environment

Using a continuous action space significantly alters the complexity of the learning process:

- **DQN**: Limited to discrete action spaces, making it unsuitable for LunarLanderContinuous without modifications such as discretization or actor-critic extensions.
- **PPO**: Well-suited for continuous control tasks, as it optimizes a parameterized stochastic policy, making smooth adjustments to the lander's controls. Our goal is to make now an algorithm that will be able to achieve the same performance for the continous LunarLander problems

Conclusion

In this project, we successfully trained reinforcement learning agents to solve the **LunarLander** environment using two different approaches: **Deep Q-Network (DQN)** for the discrete action space and **Proximal Policy Optimization (PPO)** for the continuous action space. Our experiments demonstrated that both algorithms could achieve an average score above 200, meeting the success criteria of the environment.

clement.lepretre@student-cs.fr https://github.com/ClementLptr/LunarLander-RL-Project cyrielle.theobald@student-cs.fr