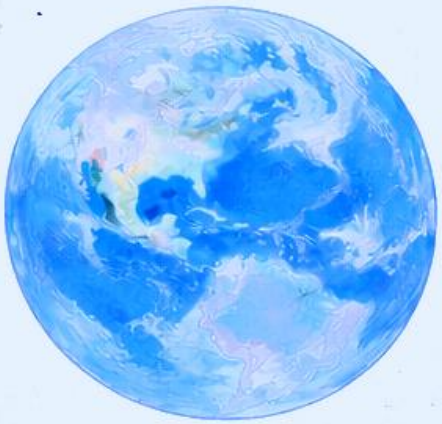
TAAP

INSTITUTO UNIVERSITÁRIO DE LISBOA



## Advanced Topics in Deep Learning

MINI-PROJECT 02 – ANNEX 02

# DEEP REINFORCEMENT LEARNING
# DQN AND PPO IN LUNAR LANDER V3

**António Cruz** (140129), **Cátia Brás** (120093), **Ricardo Kayseller** (95813)

# Table of Contents

# 1. Introduction

This annex compares the final policy quality of three different optimization approaches applied to the LunarLander-v3 environment: DQN (Deep Q-Network), PPO (Proximal Policy Optimization), and a genetic algorithm (GA). DQN and PPO are gradient-based deep reinforcement learning methods covered in the main report. The GA is a gradient-free evolutionary method included as an additional reference point.

All three methods were trained on the same environment with the same set of random seeds [42, 123, 3407]. However, their training methodologies differ substantially:

| Aspect | DQN / PPO | GA |
|---|---|---|
| Optimization | Gradient descent (backpropagation) | Evolutionary (selection, crossover, mutation) |
| Network size | 8-256-256-4 (~70k parameters) | 8-10-10-4 (244 parameters) |
| Training budget | 1.5M environment steps | ~225M environment steps |
| Gradient required | Yes | No |
| Temporal credit assignment | Yes (TD learning / advantages) | No (scalar fitness only) |

The training budgets are not directly comparable. DQN and PPO each used 1.5M environment steps with gradient-based optimization, while the GA required approximately 225M environment interactions (150x more) without any gradient computation. This difference is inherent to the respective algorithms: evolutionary methods require significantly more samples because they receive no directional feedback per weight, only a scalar fitness signal per genome.

The comparison focuses on **final policy quality** rather than sample efficiency, addressing the question: given sufficient training, how do the resulting policies compare in terms of landing performance?

```
DQN/PPO session: dqn_ppo
GA session: annex1
Seeds: [42, 123, 3407]
Evaluation episodes per seed: 20
```

## 2. Evaluation

All three algorithms are evaluated under identical conditions: 20 deterministic episodes per seed (3 seeds), for a total of 60 episodes per algorithm. The environment configuration is the same across all evaluations (LunarLander-v3, no wind). DQN and PPO use Stable-Baselines3's evaluate_policy with deterministic=True. The GA uses argmax over the neural network output, which is inherently deterministic.

### 2.1 Evaluation Summary

The tables below show per-algorithm evaluation statistics (mean, standard deviation, min, max, and success rate) broken down by seed. The overall row aggregates all 60 episodes (20 per seed x 3 seeds). A combined comparison table follows with one row per algorithm.

```
*** DQN EVALUATION SUMMARY ***
Episodes per seed: 20 | Total: 60
   Seed Mean Reward Std Dev Min Reward Max Reward Success Rate
     42      278.51   23.70     228.67     318.97       100.0%
    123      283.34   18.19     250.33     319.93       100.0%
   3407      261.22   25.88     215.37     310.74       100.0%
Overall      274.36   24.72     215.37     319.93       100.0%

*** PPO EVALUATION SUMMARY ***
Episodes per seed: 20 | Total: 60
   Seed Mean Reward Std Dev Min Reward Max Reward Success Rate
     42      278.43   14.56     251.45     311.70       100.0%
    123      270.83   14.06     242.53     292.55       100.0%
   3407      279.56   19.83     236.91     314.95       100.0%
Overall      276.27   16.81     236.91     314.95       100.0%

*** GA EVALUATION SUMMARY ***
Episodes per seed: 20 | Total: 60
   Seed Mean Reward Std Dev Min Reward Max Reward Success Rate
     42      290.64   20.52     258.55     321.98       100.0%
    123      301.73   10.95     284.02     319.78       100.0%
   3407      287.28   13.42     256.14     317.63       100.0%
Overall      293.22   16.69     256.14     321.98       100.0%

*** THREE-WAY COMPARISON ***
Seeds: [42, 123, 3407] | Episodes per seed: 20

Algorithm Mean Reward Std Dev Min Reward Max Reward Success Rate   Network Env Steps
      DQN      274.36   24.72     215.37     319.93       100.0% 2x256 MLP      1.5M
      PPO      276.27   16.81     236.91     314.95       100.0% 2x256 MLP      1.5M
       GA      293.22   16.69     256.14     321.98       100.0%  2x10 MLP     ~225M
```

## 2.2 Comparison Charts

The following chart presents a grouped bar comparison of mean reward per seed across all three algorithms. Error bars represent one standard deviation. This visualization allows direct comparison of per-seed performance and cross-seed consistency within each algorithm.

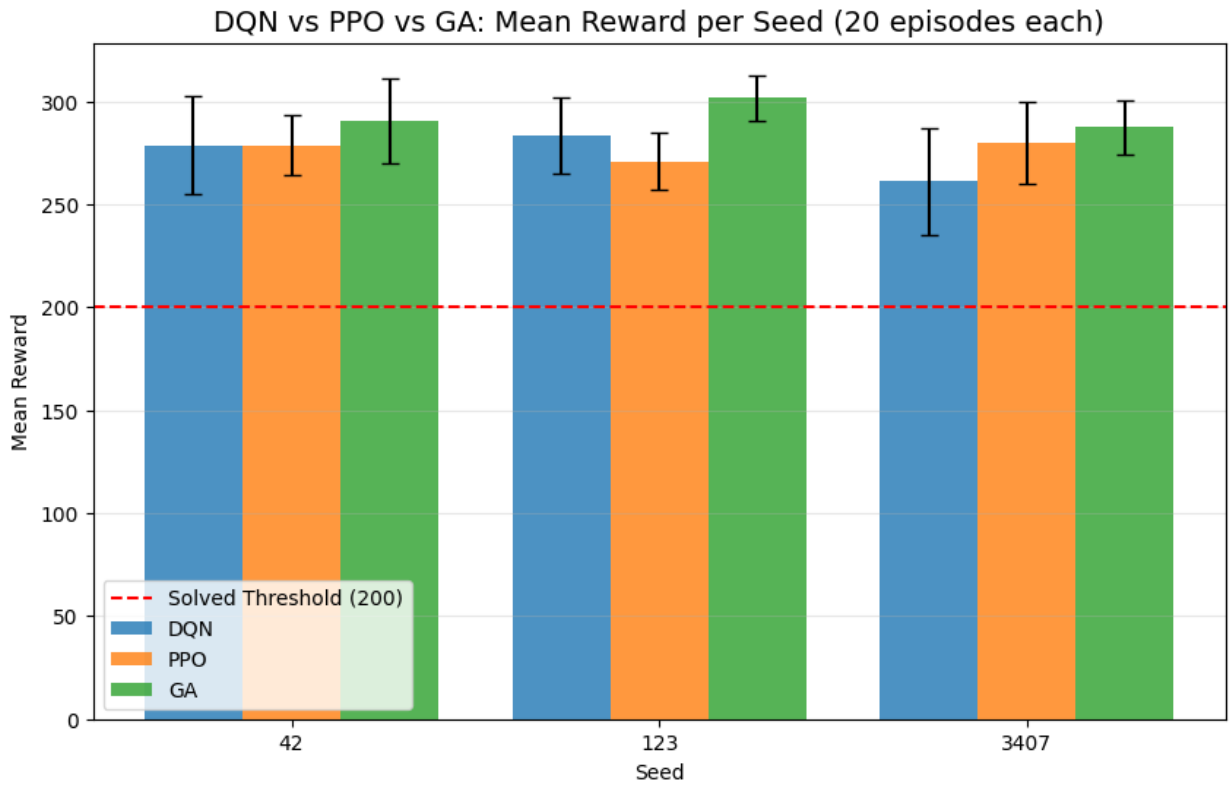



**Figure 1** - Mean evaluation reward per seed for DQN, PPO, and GA, with standard deviation error bars. All three algorithms exceed the solved threshold (200) across all seeds.

The next chart collapses all seeds into a single overall mean reward per algorithm, providing a high-level comparison of aggregate performance.

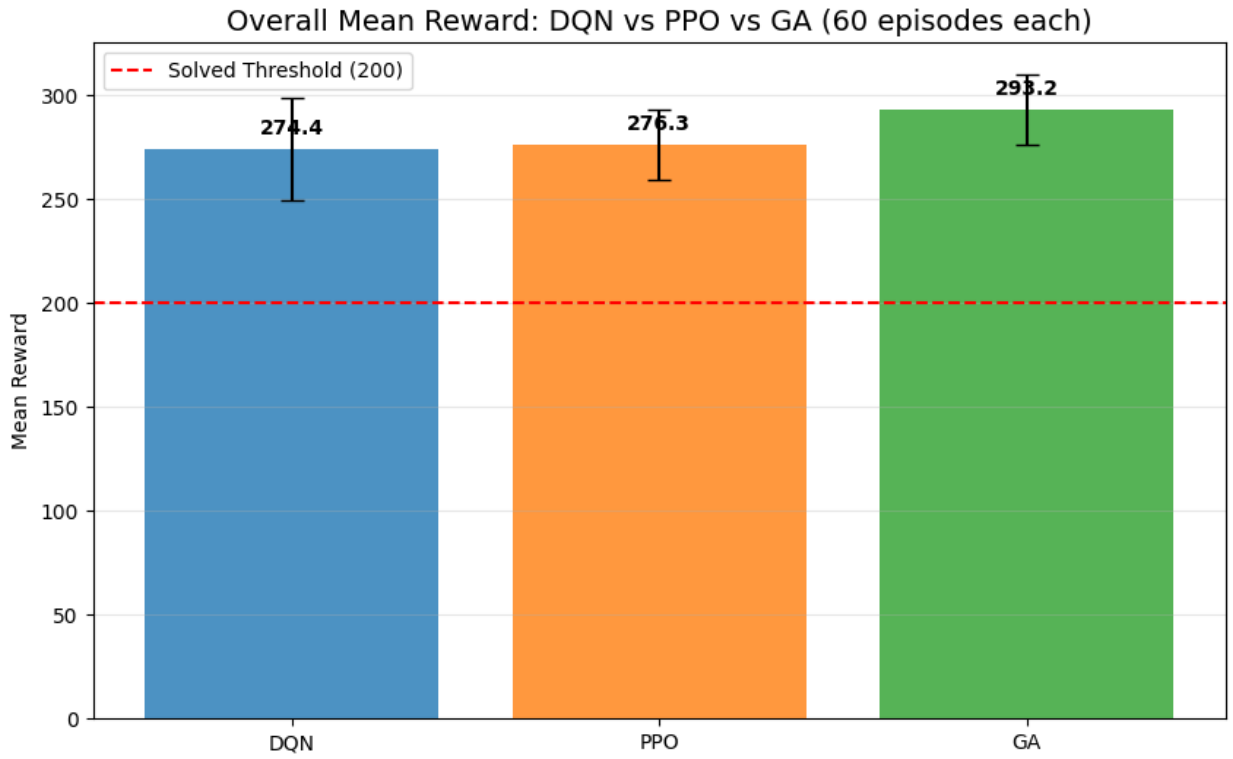



**Figure 2** - Overall mean evaluation reward per algorithm, aggregated across all seeds and episodes (60 episodes each). All three approaches achieve mean rewards well above the solved threshold.

The overlaid histogram below shows the full reward distribution for each algorithm, combining all seeds. This reveals the spread, overlap, and shape of the reward distributions beyond what summary statistics capture.

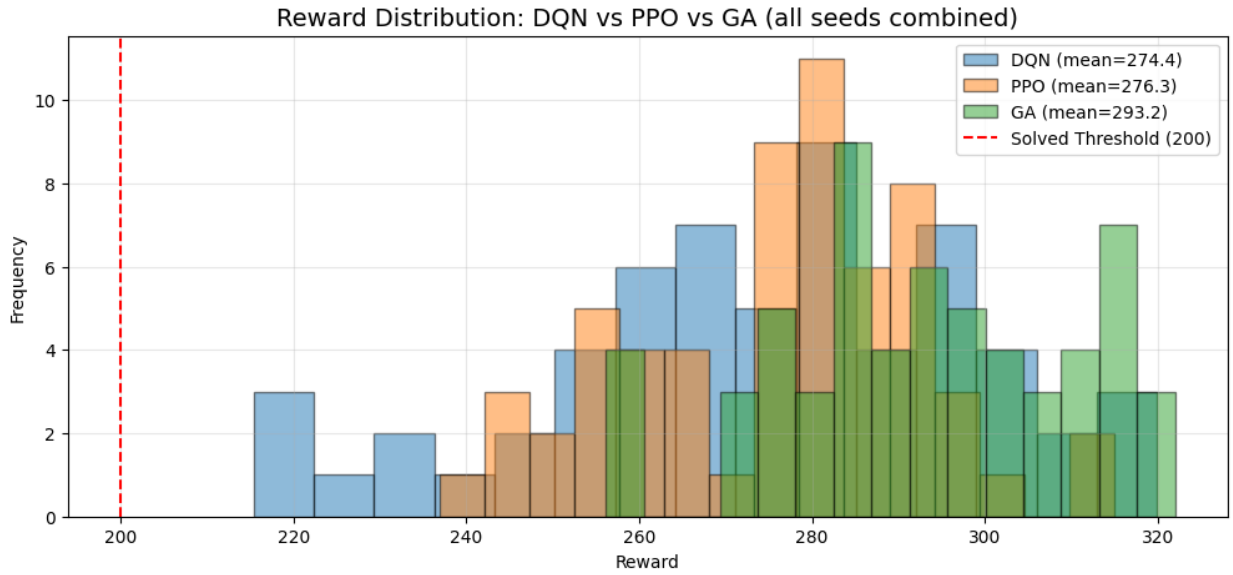



**Figure 3** - Overlaid reward distributions for DQN, PPO, and GA across all seeds combined, showing the spread and overlap of evaluation outcomes.

Box plots provide a complementary view of the reward distributions, showing medians, quartiles, and outliers for each algorithm broken down by seed.

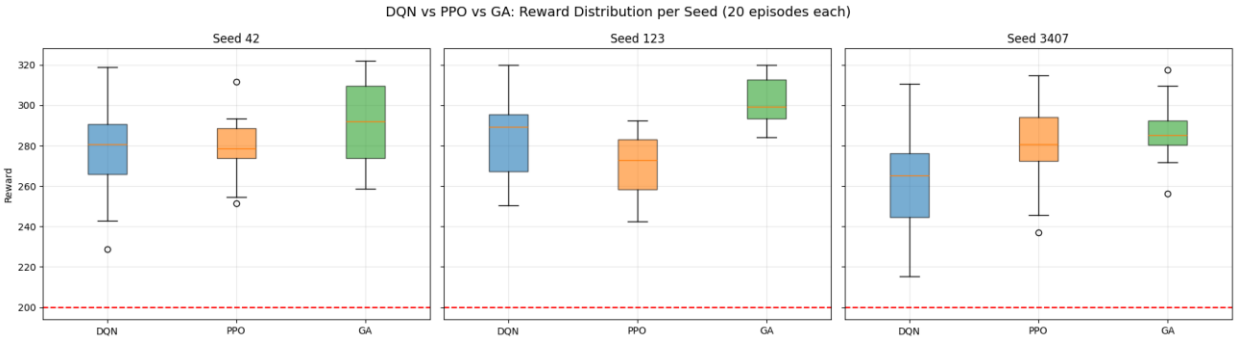



**Figure 4** - Box plot comparison of reward distributions per seed for DQN, PPO, and GA. The boxes show the interquartile range, with whiskers extending to 1.5x IQR.

## 3. Statistical Significance

To determine whether observed performance differences are statistically meaningful, pairwise Mann-Whitney U tests compare the reward distributions between all algorithm pairs (DQN vs PPO, DQN vs GA, PPO vs GA). Chi-squared tests compare success rates (proportion of episodes achieving reward >= 200). With all three algorithms achieving high success rates, the Chi-squared tests may lack discriminative power; the Mann-Whitney U tests on raw rewards provide a more sensitive comparison.

```
*** PAIRWISE REWARD COMPARISON (Mann-Whitney U) ***
Sample size per algorithm: 60 episodes

Comparison Mean 1 Mean 2 U Statistic p-value Significant
DQN vs PPO 274.36 276.27      1771.0  0.8811          No
 DQN vs GA 274.36 293.22       987.0  0.0000         Yes
 PPO vs GA 276.27 293.22       886.0  0.0000         Yes

*** PAIRWISE SUCCESS RATE COMPARISON (Chi-squared) ***

Comparison Success 1 Success 2                 Test Statistic p-value Significant
DQN vs PPO    100.0%    100.0% Chi-squared (skipped)      0.00  1.0000          No
 DQN vs GA    100.0%    100.0% Chi-squared (skipped)      0.00  1.0000          No
 PPO vs GA    100.0%    100.0% Chi-squared (skipped)      0.00  1.0000          No
```

## 4. Agent Behavior Analysis

Beyond aggregate reward metrics, examining how each algorithm controls the lander reveals differences in learned strategies. The action distribution shows the relative frequency of each discrete action (do nothing, fire left, fire main, fire right) across all evaluation episodes. Different algorithms may develop distinct control patterns — for example, varying reliance on the main thruster versus lateral corrections.

```
DQN: 14,451 total actions across 60 episodes
PPO: 16,544 total actions across 60 episodes
GA: 11,271 total actions across 60 episodes

Behavior data collection complete.
```

The grouped bar charts below compare action usage in both absolute counts and percentage terms across all three algorithms.

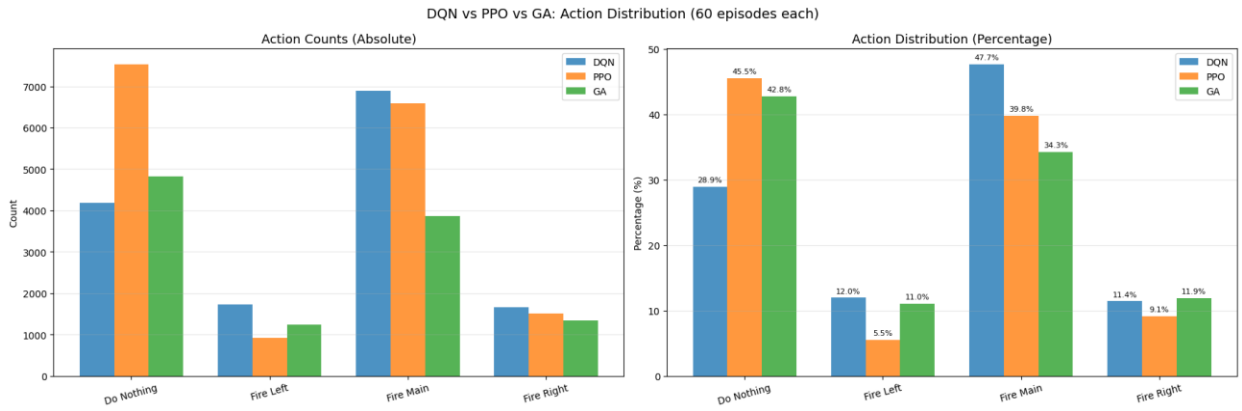



**Figure 5** - Action distribution for DQN, PPO, and GA across all evaluation episodes, showing both absolute counts and relative percentages for each of the four discrete actions.

Trajectory plots show the x-y path of the lander from launch to landing for a sample of episodes. Each algorithm's trajectories are shown separately, followed by a direct overlay on the same chart for comparison.

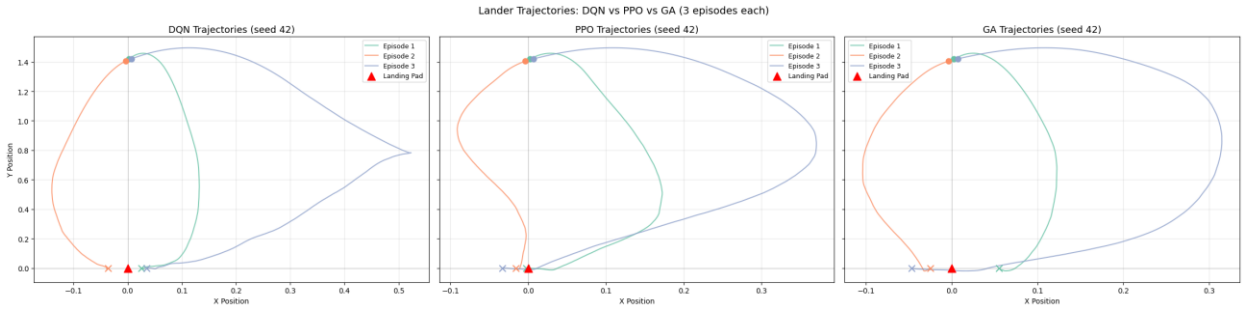



**Figure 6** - Sample landing trajectories (x-y position) for each algorithm, showing the spatial path from launch to touchdown. Circles mark the start position, crosses mark the end position.

The overlay below places one trajectory from each algorithm on the same axes, making it easier to compare the flight profiles directly.

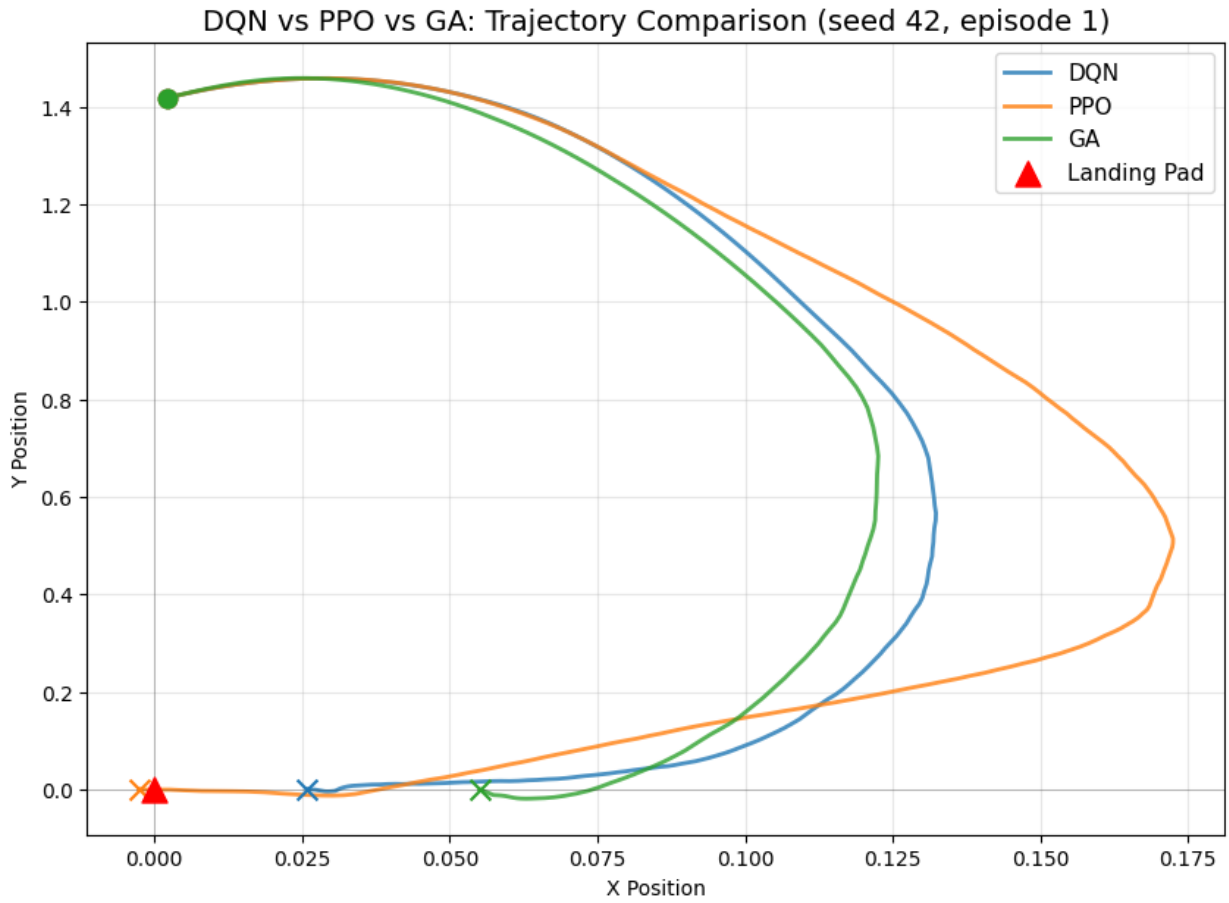
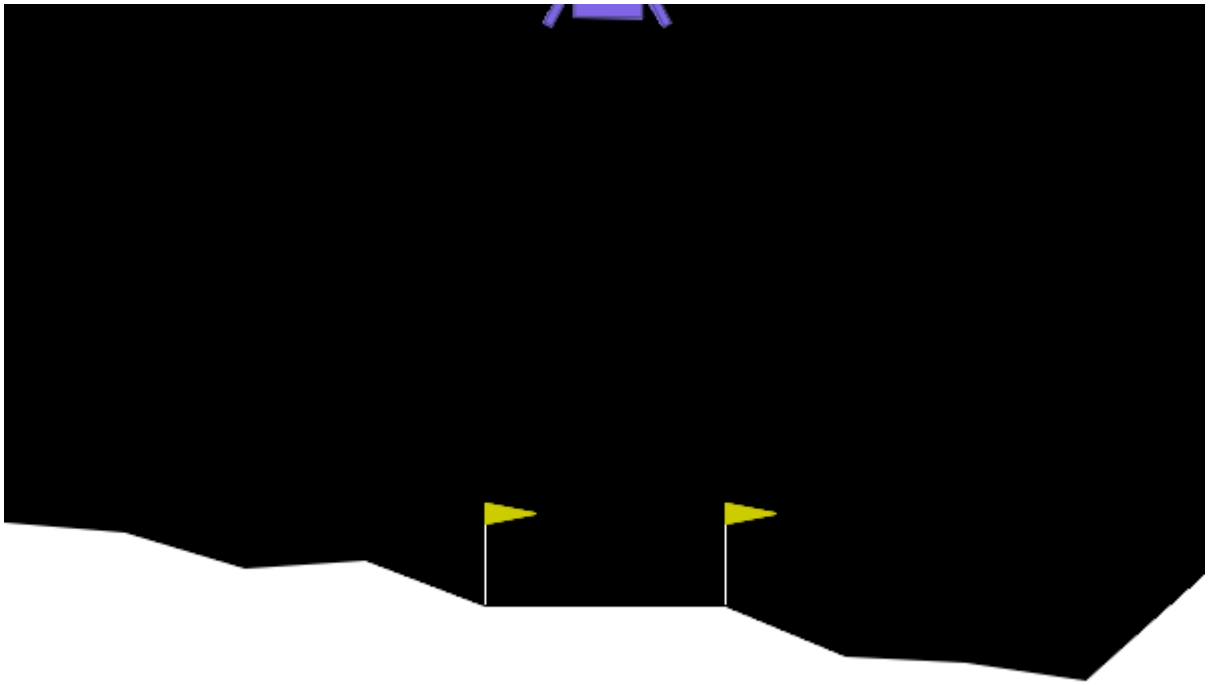


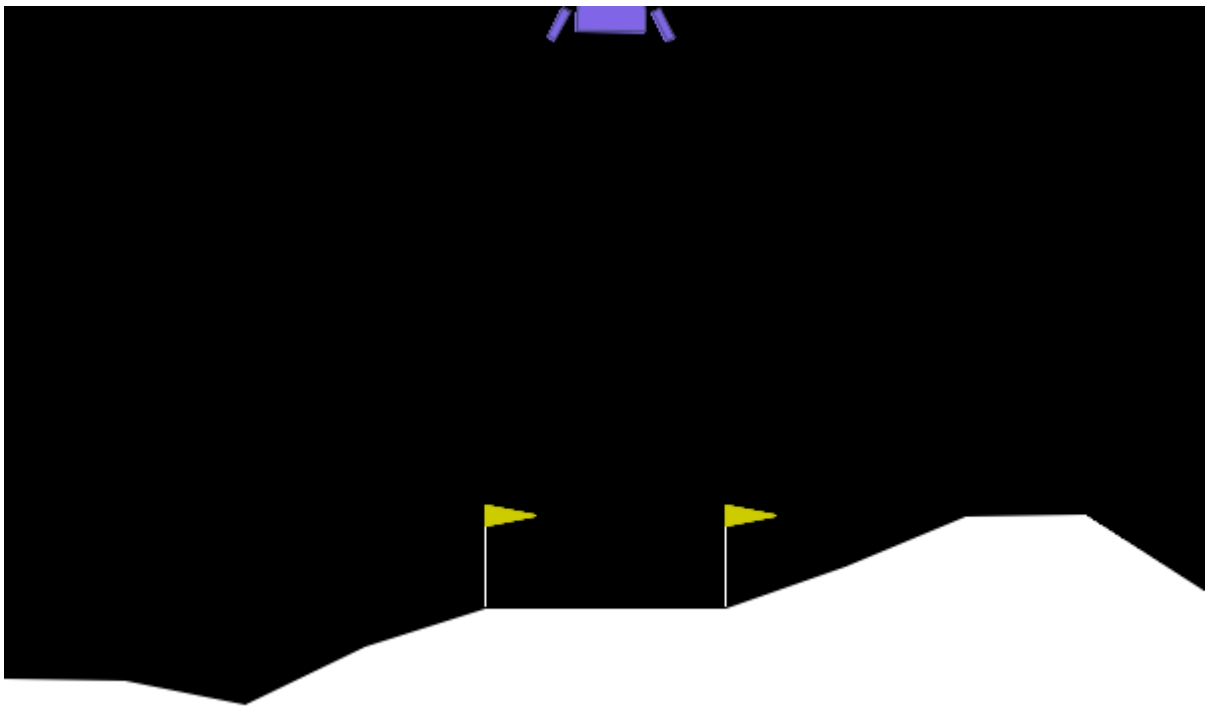

**Figure 7** - Direct overlay of one landing trajectory per algorithm on the same chart, illustrating differences in flight profile and approach strategy.

## 5. Visualizations

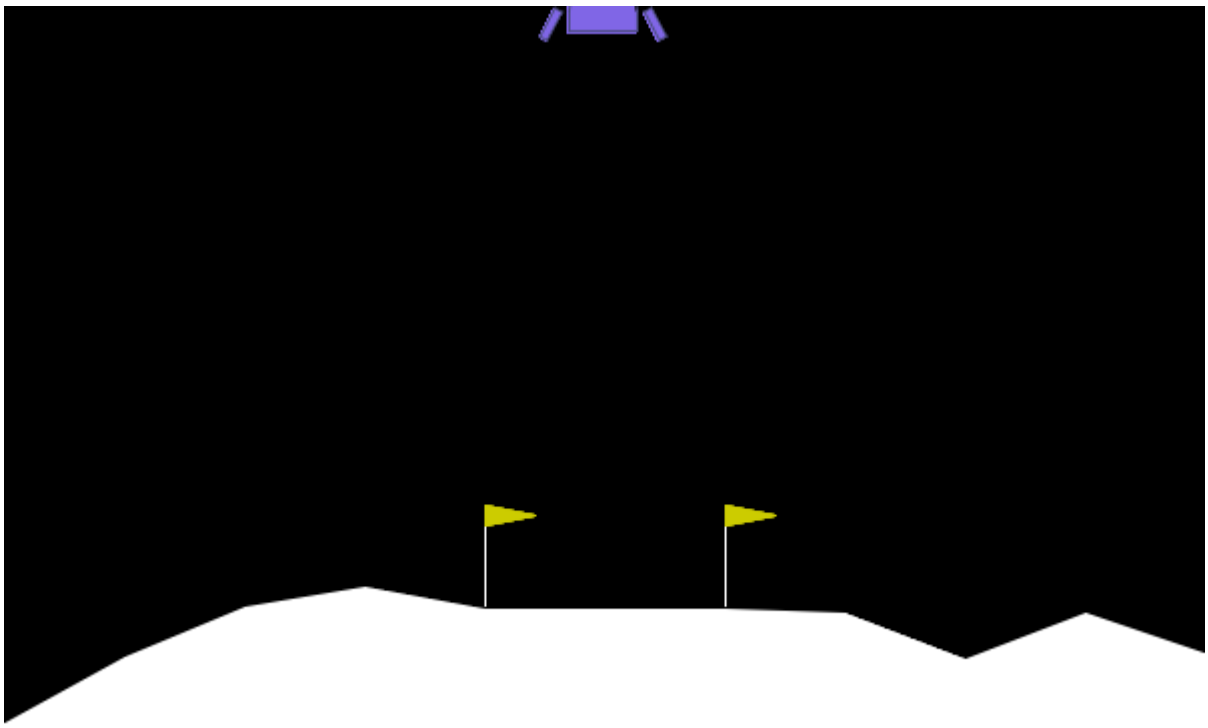
The animated GIFs below show a single deterministic episode for each algorithm and seed, providing a visual impression of the learned landing behaviour. Each GIF renders the full episode from launch to termination.



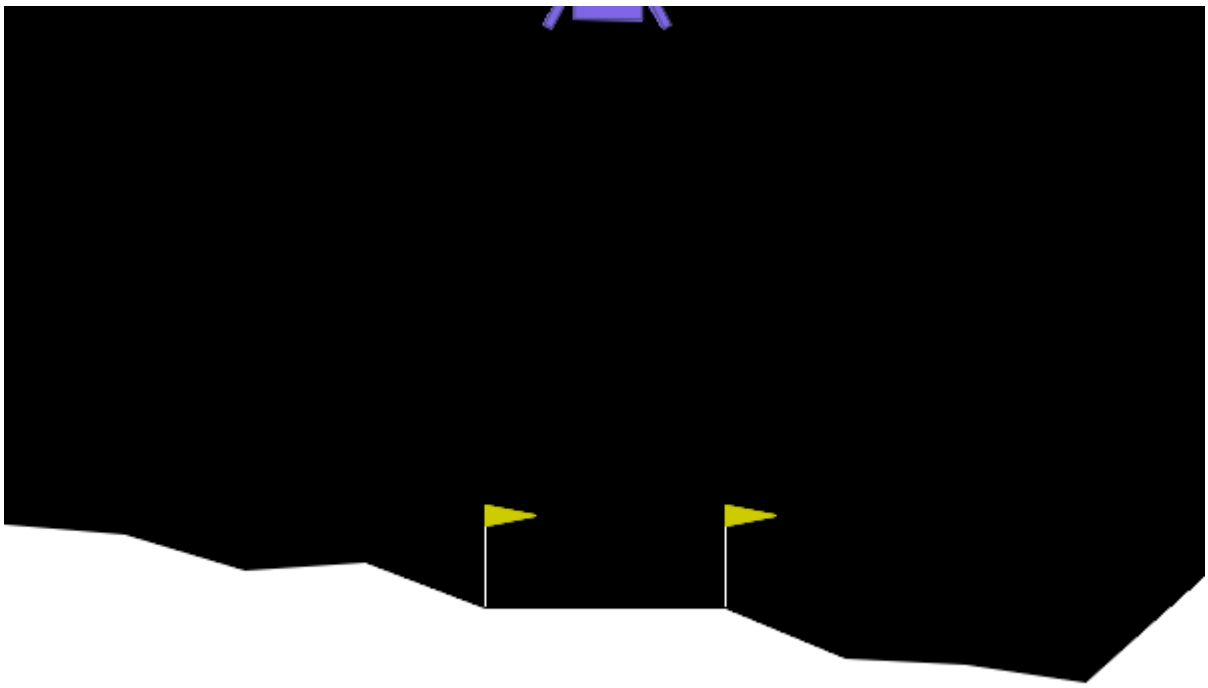
**Gif 1** - DQN Landing (Seed 42)



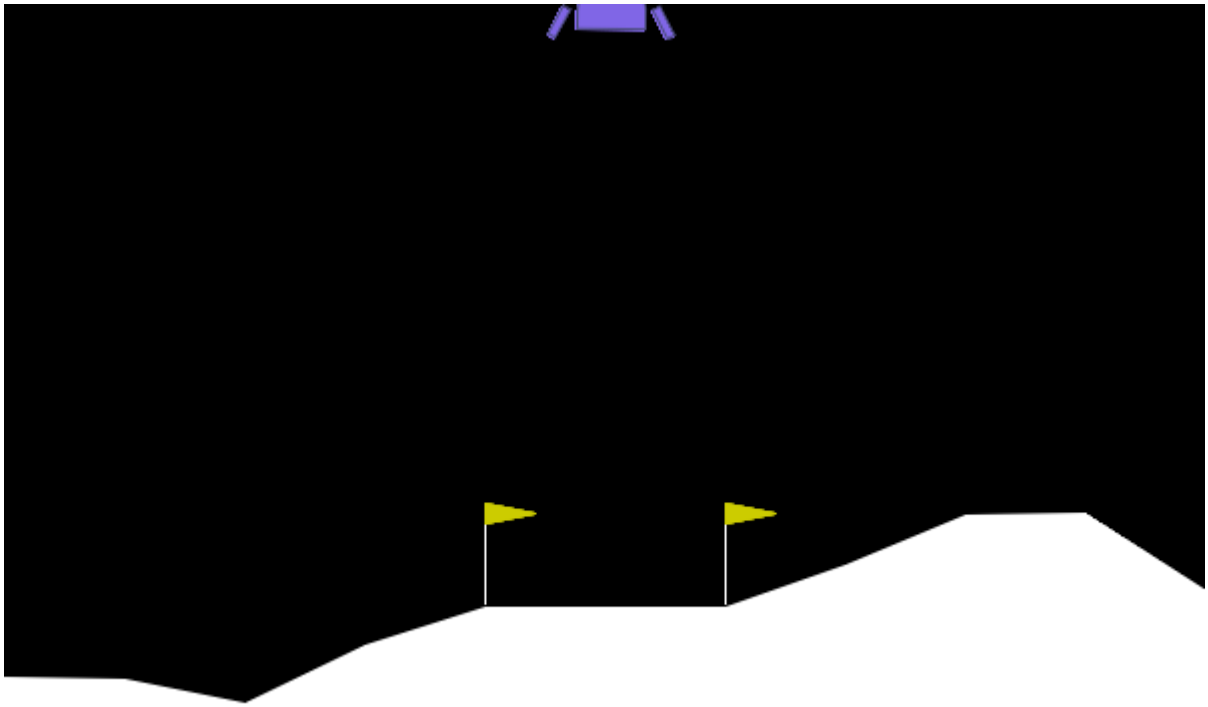
**Gif 2** - DQN Landing (Seed 123)



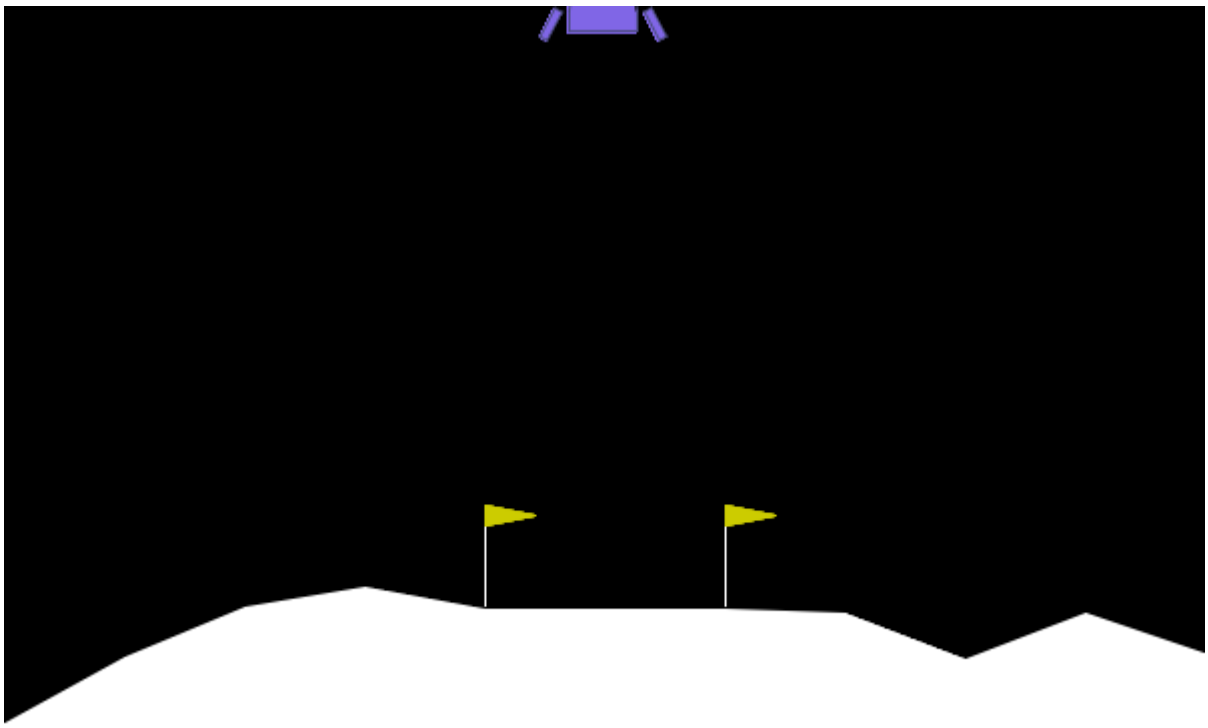
**Gif 3** - DQN Landing (Seed 3407)



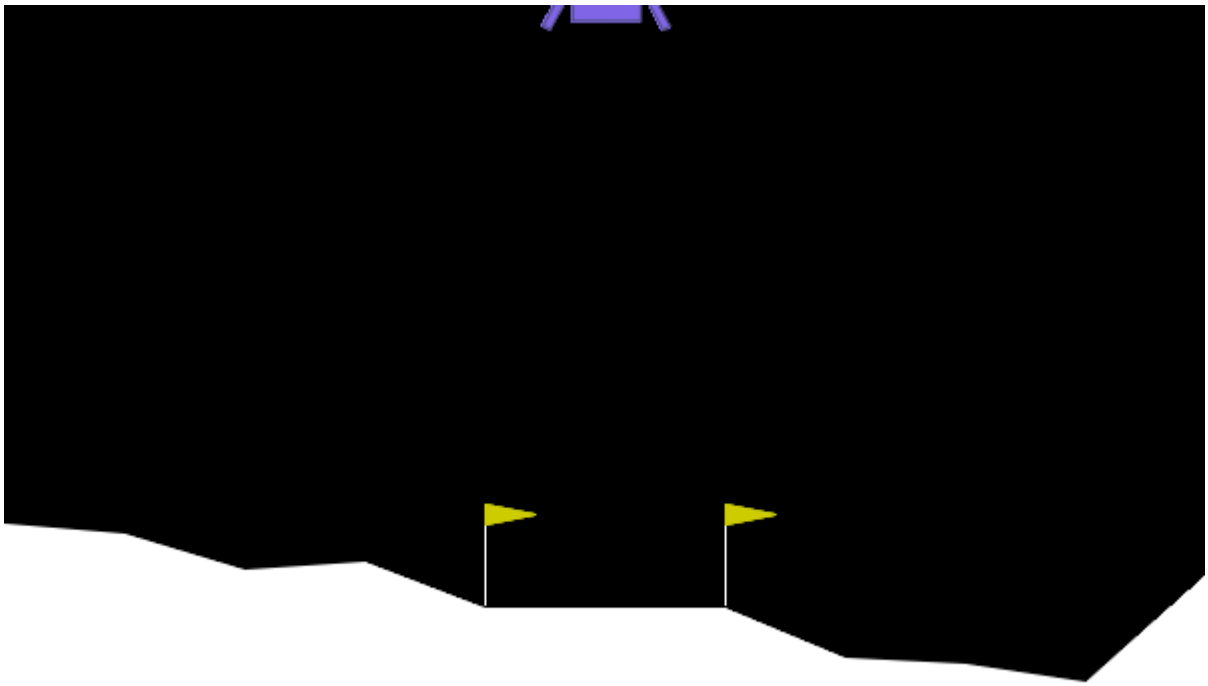
**Gif 4** - PPO Landing (Seed 42)



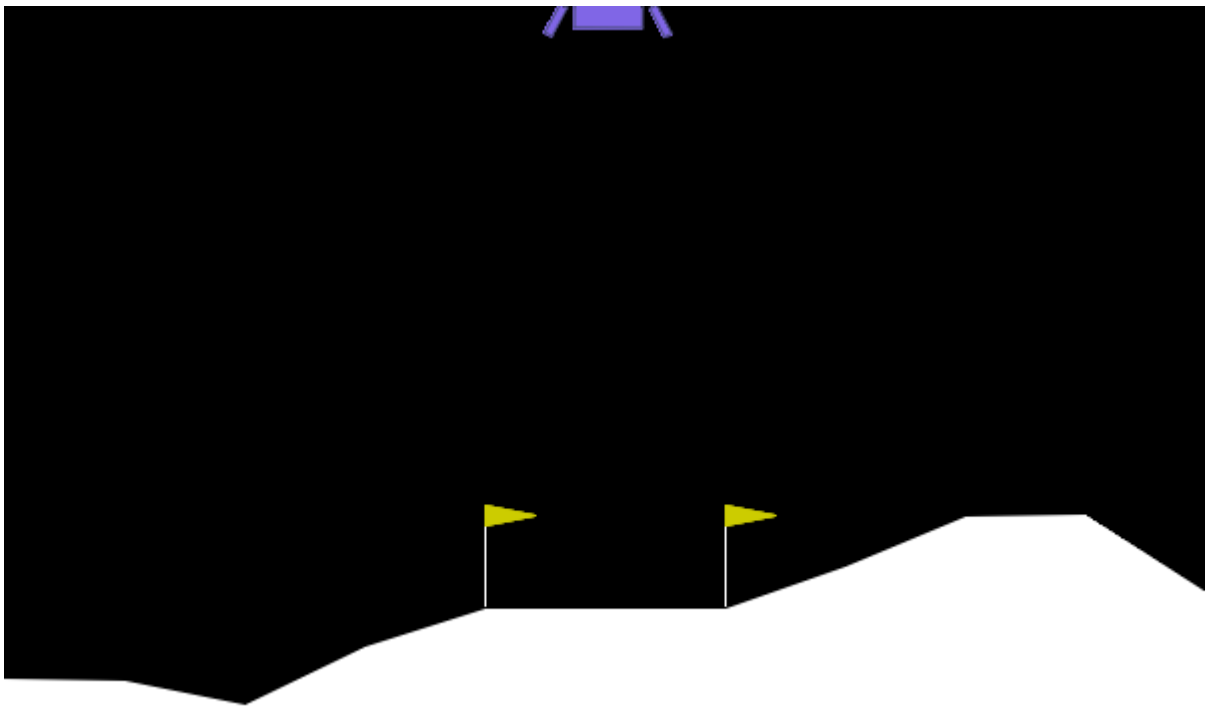
**Gif 5** - PPO Landing (Seed 123)



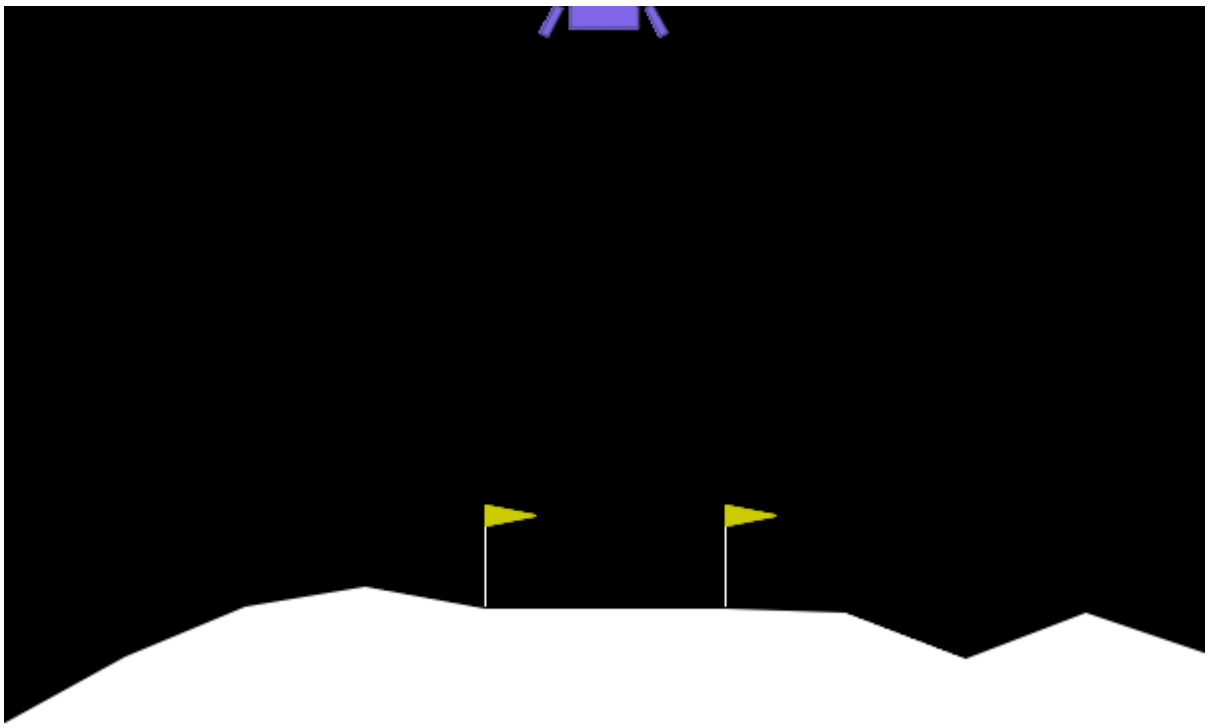
**Gif 6** - PPO Landing (Seed 3407)



**Gif 7** - GA Landing (Seed 42)



**Gif 8** - GA Landing (Seed 123)



**Gif 9** - GA Landing (Seed 3407)

## 6. Conclusions

This annex compared the final policy quality of three optimization approaches — DQN, PPO, and a genetic algorithm — evaluated under identical conditions on the LunarLander-v3 environment.

All three algorithms successfully solved the environment, achieving 100% success rates across all seeds. DQN achieved a mean reward of 274.36, PPO achieved 276.27, and the GA achieved 293.22. While all three approaches produced effective landing policies, the observed performance differences should be interpreted in context:

- **DQN and PPO** are gradient-based deep RL methods that learned effective policies using 1.5M environment steps each, with large network architectures (~70k parameters). These methods exploit the temporal structure of experience — sequences of states, actions, and rewards — to efficiently assign credit to individual decisions.
- **The GA** used a much smaller network (244 parameters) but required approximately 225M environment steps (150x more) to achieve its results. It has no mechanism for temporal credit assignment and relies entirely on scalar fitness signals.

The fact that all three approaches converge to high-quality policies on this environment reflects the characteristics of LunarLander-v3 itself: an 8-dimensional continuous observation space, 4 discrete actions, dense reward shaping, and short episodes. These properties make the problem tractable for diverse optimization approaches.

The summary table below captures the key dimensions of each approach.

```
*** ALGORITHM COMPARISON SUMMARY ***
Algorithm                         Network                 Training     Mean Reward  Success
DQN  Gradient-based (off-policy) 8-256-256-4 (MLP) ~70k  1.5M env steps    274.36   100.0%
PPO  Gradient-based (on-policy)  8-256-256-4 (MLP) ~70k  1.5M env steps    276.27   100.0%
GA Evolutionary (gradient-free)  8-10-10-4 (MLP)   244   ~225M env steps   293.22   100.0%
```

The table above summarizes the key dimensions of each approach. The reported performance reflects evaluation on 20 deterministic episodes per seed under identical environment conditions, ensuring a fair comparison of final policy quality regardless of the different training methodologies.

The sample efficiency advantage of DQN and PPO (1.5M vs ~225M environment steps) is a fundamental property of gradient-based optimization: by computing how each weight should change based on temporal feedback, these methods converge with far fewer environment interactions. This advantage becomes increasingly significant in environments where interactions are costly, high-dimensional, or involve continuous action spaces - precisely the settings where deep reinforcement learning methods are designed to excel.