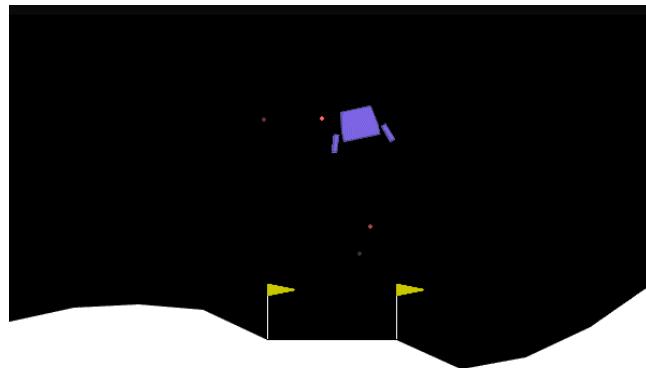


Εργασία 9 – Βαθιά Ενισχυτική Μάθηση

Στόχος της εργασίας είναι η εκπαίδευση ενός πράκτορα που χειρίζεται την προσγείωση ενός διαστημοπλοίου (Lunar Lander). Πιο συγκεκριμένα, το διαστημόπλοιο πρέπει να προσγειωθεί σε ένα σε συγκεκριμένο ομαλό σημείο πάνω στο φεγγάρι.



Αν και πρόκειται για παιχνίδι, το περιβάλλον παρέχει αρκετές επιλογές για την προσομοίωση διαφόρων νόμων της φυσικής, οι οποίοι μπορούν να παραμετροποιηθούν (πχ Δύναμη της τουρμπίνας, σταθερότητα κινητήρα, δύναμη της βαρύτητας, κλπ.) Περισσότερες λεπτομέρειες θα βρείτε στον παρακάτω σύνδεσμο:

https://www.gymlibrary.dev/environments/box2d/lunar_lander/

Για την υλοποίηση του πράκτορα θα χρησιμοποιηθεί η βιβλιοθήκη `rllib` και για το περιβάλλον η βιβλιοθήκη `gymnasium`.

Οδηγίες:

1. Εγκαταστήστε (`!pip install`) τις 4 βιβλιοθήκες με τη σειρά (σε ξεχωριστά κελιά): `swig`, `gymnasium[box2d]`, `renderlab`
2. Κάθε περιβάλλον `gymnasium` αποτελείτε από 3 βασικές συναρτήσεις: `step`, `reset`, `render`. Στη συνέχεια, να περιγράψετε τη λειτουργία της κάθε μίας. Μπορείτε να συμβουλευτείτε το documentation https://www.gymlibrary.dev/content/basic_usage/
3. Να περιγράψετε το περιβάλλον `LunarLander-v3` ως εξής:
 - a. Observation Space
 - b. Action Space
 - c. Reward Function
4. Δημιουργήστε έναν `random agent` και χρησιμοποιήστε τη βιβλιοθήκη `renderlab` ώστε να οπτικοποιήσετε τον τρόπο παιχνιδιού του (όπως φαίνεται στην παρακάτω εικόνα).
5. Τρέξτε τον `random agent` για 5 επεισόδια και στη συνέχεια υπολογίστε το μέσο score που πετυχαίνει.
6. Χρησιμοποιήστε τη βιβλιοθήκη [Stable Baselines](#) για να εκπαιδεύσετε τους αλγορίθμους για συγκεκριμένο αριθμό βημάτων της επιλογής σας:
 - a. DQN

b. PPO

7. Υπολογίστε το μέσο reward σε 5 επεισόδια για καθένα από τους πράκτορες και δημιουργήστε γραφήματα που συγκρίνεται τους αλγορίθμους σε 1) Χρόνο εκπαίδευσης 2) Rewards ανά βήμα/επεισόδιο (κατά την εκπαίδευση)

The screenshot shows a Jupyter Notebook cell with the following code:

```
1 import renderlab as rl
2
3 env = gym.make("CartPole-v1", render_mode = "rgb_array")
4 env = rl.RenderFrame(env, "./output")
5
6 observation, info = env.reset()
7
8 while True:
9     action = env.action_space.sample() # Replace this with the action selected by the agent
10    observation, reward, terminated, truncated, info = env.step(action)
11
12    if terminated or truncated:
13        break
14
15 env.play()
```

Below the code, there is a terminal output:

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_and_should_run_async(code)
WARNING:py.warnings:/usr/local/lib/python3.10/dist-packages/moviepy/video/fx/painting.py:7: De
from scipy.ndimage.filters import sobel

Moviepy - Building video temp-{start}.mp4.
Moviepy - Writing video temp-{start}.mp4

Moviepy - Done !
Moviepy - video ready temp-{start}.mp4
```

At the bottom, there is a video player interface showing a simulation of a cart-pole system. The cart is black and the pole is brown. The video player controls show 0:00 / 0:00.

8. Επαναλάβετε τα ερωτήματα 6,7 βρίσκοντας όμως κατάλληλες παραμέτρους για τους DQN, PPO στο LunarLander, κάνοντας αναζήτηση στο διαδίκτυο.