Deep RL with MuJoCo

Adrià de Angulo

Daniel Matas

Hariss Mohammad Jabeen

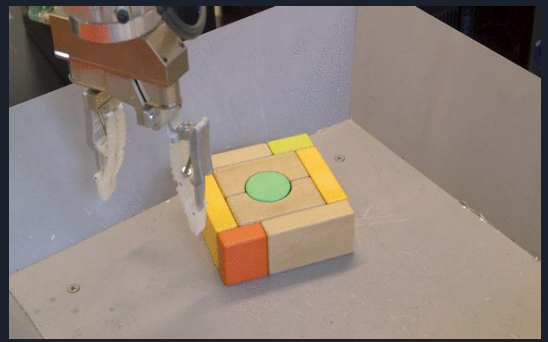
Miquel Quesada

Advisor: JuanJo Nieto

https://github.com/danimatasd/MUJOCO-AIDL

Motivation of the project

- Appliance of RL to solve dynamic /real world problems (robotics, autonomous driving, healthcare…)
- Trial and error approach of the RL process
- Deployment of RL algorithms in dynamic environments generated by a virtual engine (MuJoCo)



Goals

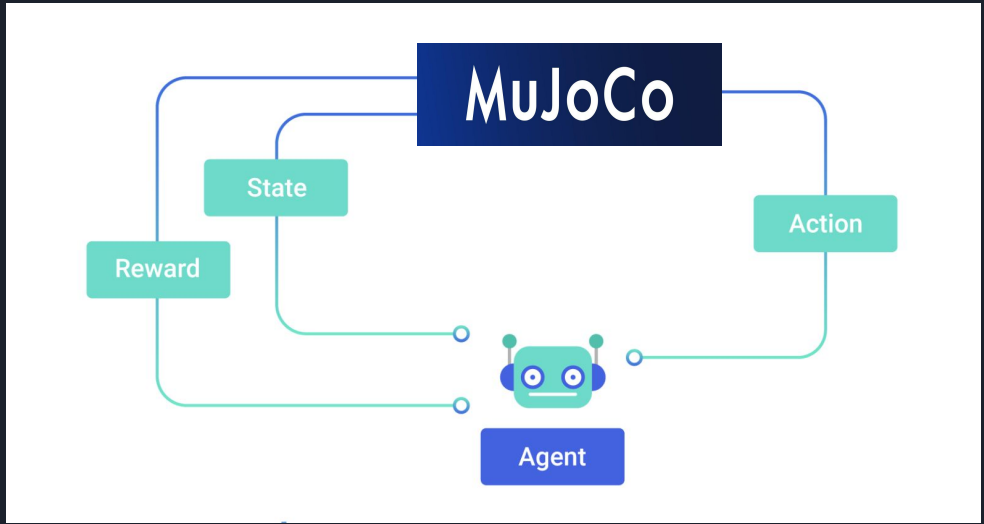
Learn about the basics of RL (states, actions, policies…) and use a physics engine to produce an accurate simulation

Make the robot be able to walk in one direction

Make the robot be able to walk over a small step without falling over

Apply the concepts of MLOps to deploy a replicable repository that can be runned by any user

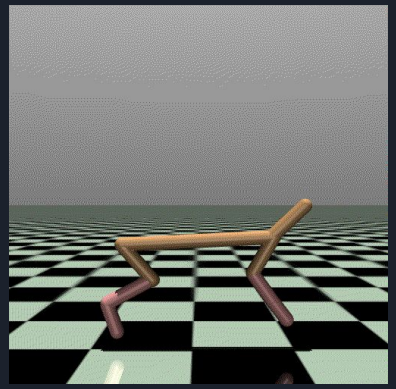
Our proposal: Data



Source: Perfectial

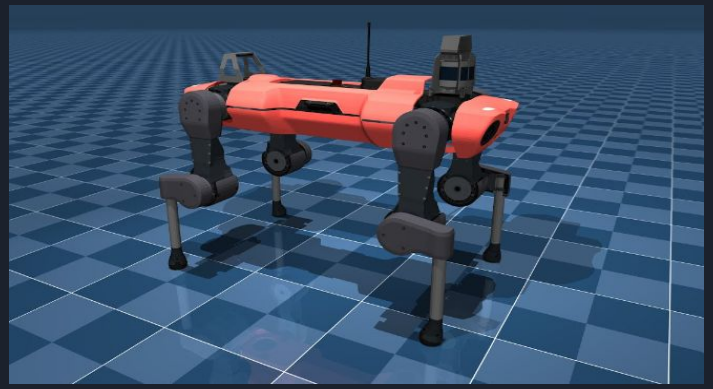
Our proposal: Environment

Half Cheetah



Gym

Anymal C



Custom with Mujoco Engine

Our proposal: Computational Resources

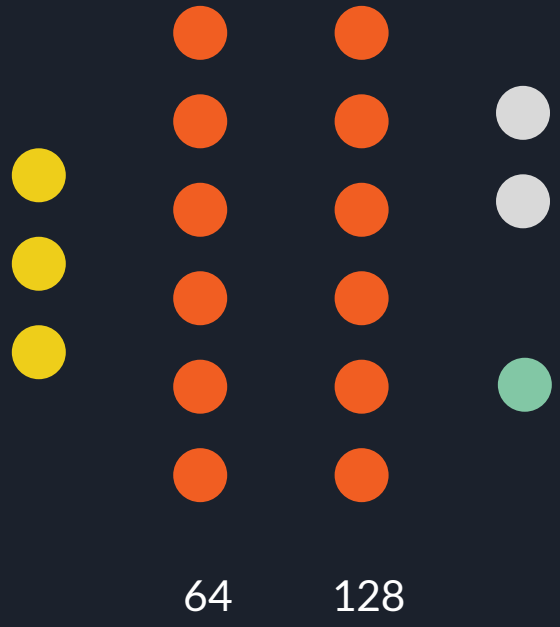
Algorithm 1 PPO, Actor-Critic Style

for iteration=$1, 2, \ldots$ **do**
for actor=$1, 2, \ldots, N$ **do**
Run policy $\pi_{\theta_{\text{old}}}$ in environment for T timesteps
Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
end for
Optimize surrogate L wrt θ, with K epochs and minibatch size $M \leq NT$
$\theta_{\text{old}} \leftarrow \theta$
end for

Source: OpenAI

Our proposal: Half Cheetah NN

17 Half Cheetah Environment Inputs

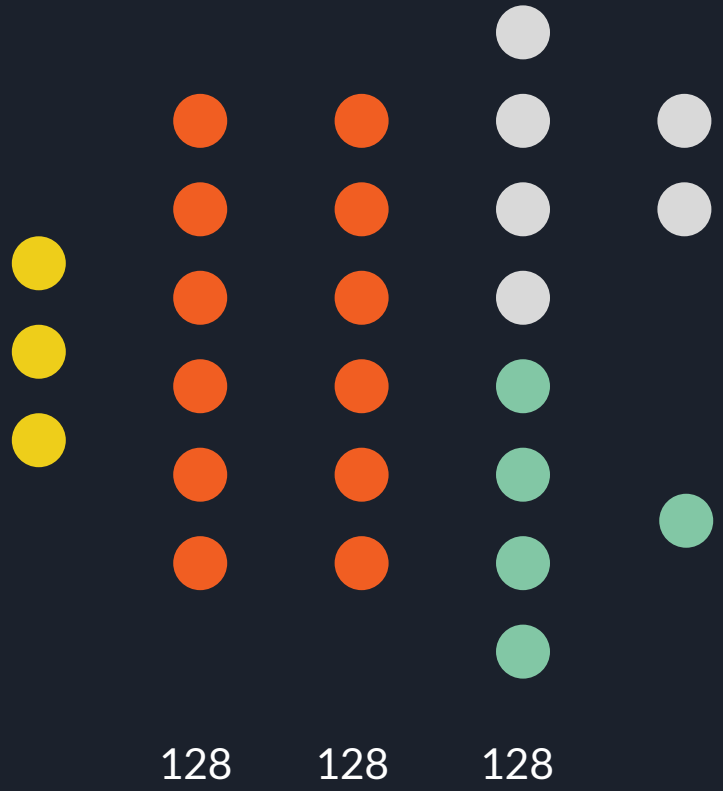


6 Torque Outputs

1 Expected Return

Our proposal: Anymal C NN

37 Anymal C
Environment Inputs

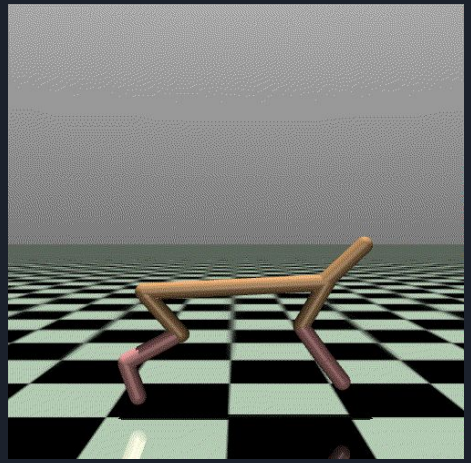


12 Position Outputs

1 Expected Return

Milestones

1) Learn the theoretical base of RL ✓
2) Establish a teamwork setup ✓
3) Get familiar with the engine ✓
4) Generate a “base” code to train a model ✓
5) + Apply code to Half Cheetah
6) + Create a Mujoco Environment function similar to Gym
7) + Apply code to ANYmal C
8) + Hyperparameter Sweep for each code
9) + Final run for each code
10) + Get videos for each run

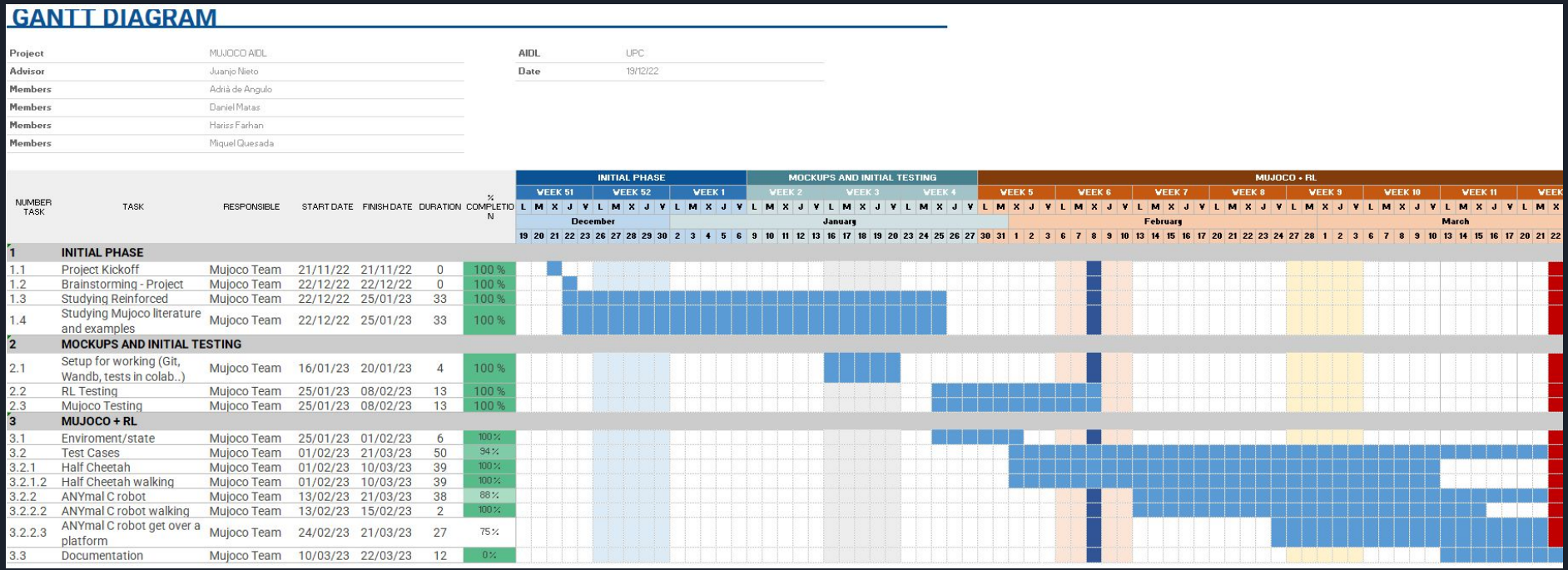


Project Plan

GANTT DIAGRAM

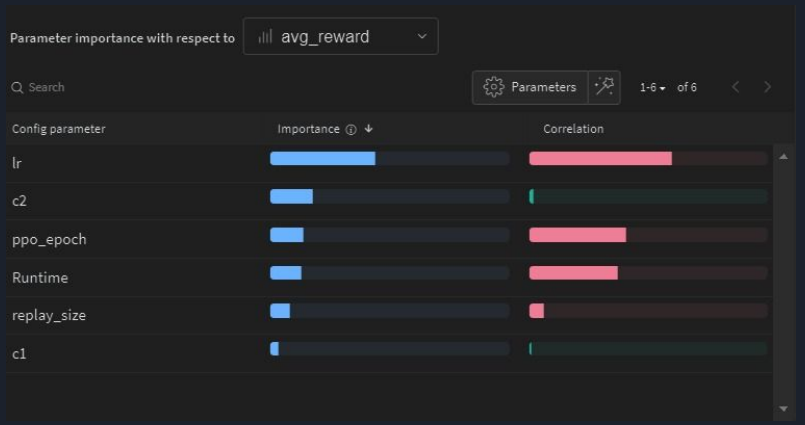
Project	MUJOCO AIDL
Advisor	Juanjo Nieto
Members	Adrià de Angulo
Members	Daniel Matas
Members	Hariss Farhan
Members	Miquel Quesada

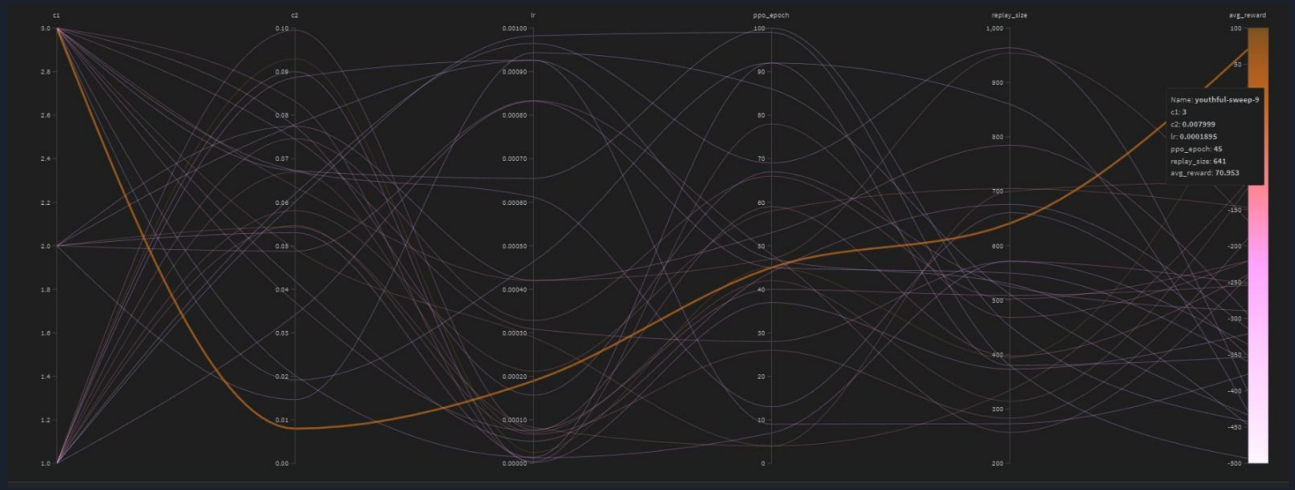
AIDL	UPC
Date	19/12/22



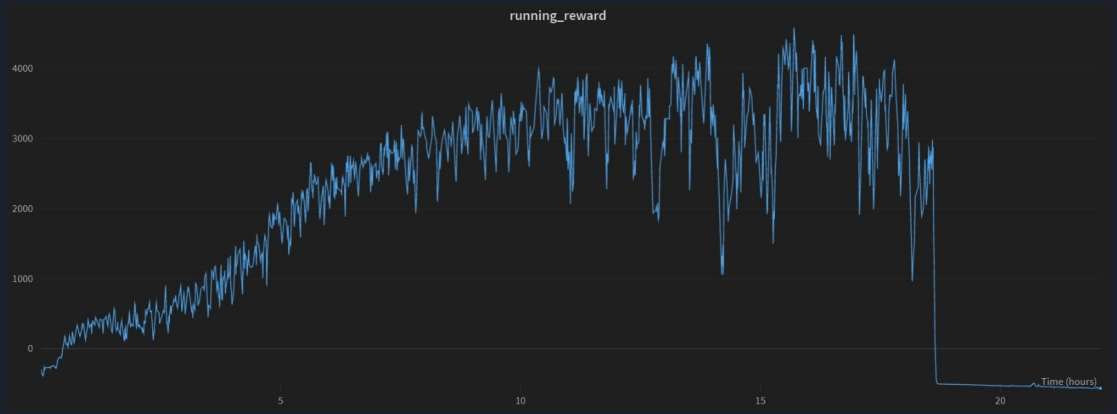
NUMBER TASK	TASK	RESPONSIBLE	START DATE	FINISH DATE	DURATION	% COMPLETION
1	**INITIAL PHASE**					
1.1	Project Kickoff	Mujoco Team	21/11/22	21/11/22	0	100 %
1.2	Brainstorming - Project	Mujoco Team	22/12/22	22/12/22	0	100 %
1.3	Studying Reinforced	Mujoco Team	22/12/22	25/01/23	33	100 %
1.4	Studying Mujoco literature and examples	Mujoco Team	22/12/22	25/01/23	33	100 %
2	**MOCKUPS AND INITIAL TESTING**					
2.1	Setup for working (Git, Wandb, tests in colab..)	Mujoco Team	16/01/23	20/01/23	4	100 %
2.2	RL Testing	Mujoco Team	25/01/23	08/02/23	13	100 %
2.3	Mujoco Testing	Mujoco Team	25/01/23	08/02/23	13	100 %
3	**MUJOCO + RL**					
3.1	Enviroment/state	Mujoco Team	25/01/23	01/02/23	6	100%
3.2	Test Cases	Mujoco Team	01/02/23	21/03/23	50	94%
3.2.1	Half Cheetah	Mujoco Team	01/02/23	10/03/23	39	100%
3.2.1.2	Half Cheetah walking	Mujoco Team	01/02/23	10/03/23	39	100%
3.2.2	ANYmal C robot	Mujoco Team	13/02/23	21/03/23	38	88%
3.2.2.2	ANYmal C robot walking	Mujoco Team	13/02/23	15/02/23	2	100%
3.2.2.3	ANYmal C robot get over a platform	Mujoco Team	24/02/23	21/03/23	27	75%
3.3	Documentation	Mujoco Team	10/03/23	22/03/23	12	0%

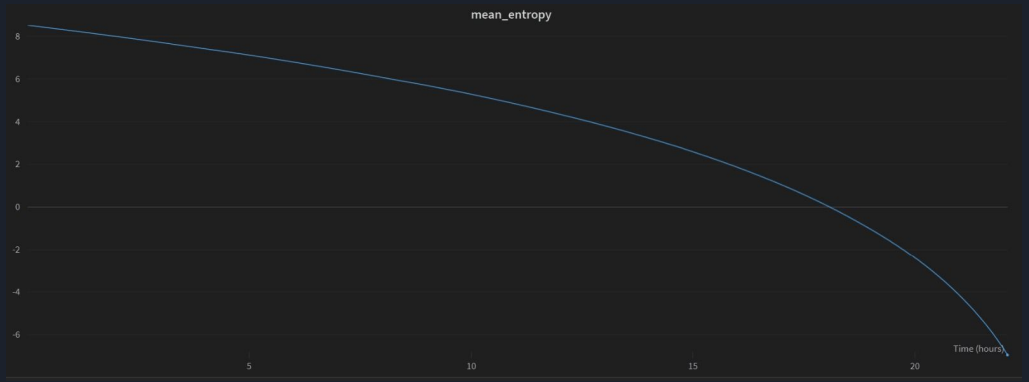
Results Half-Cheetah





Results Half-Cheetah





Results Half-Cheetah

-304

1000

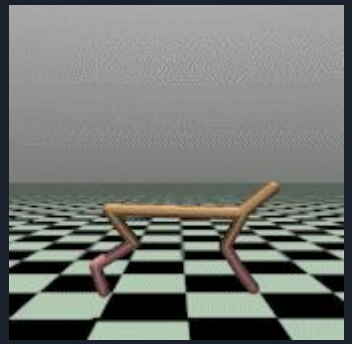
2000

3000

3908

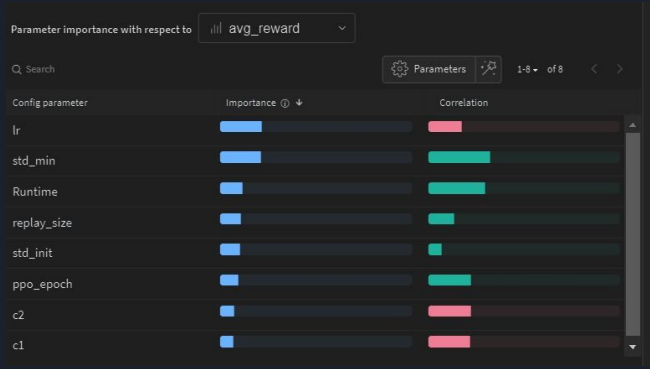
5006

5734

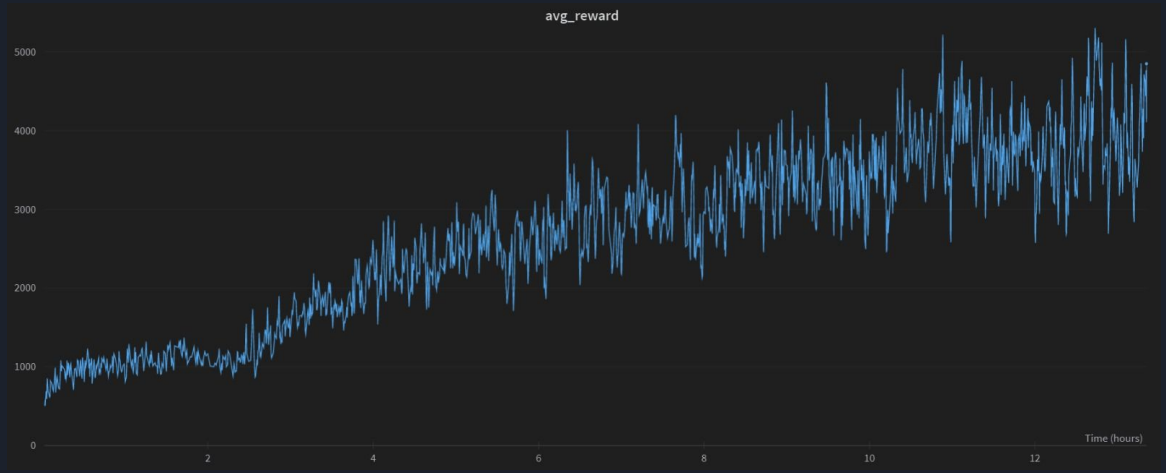


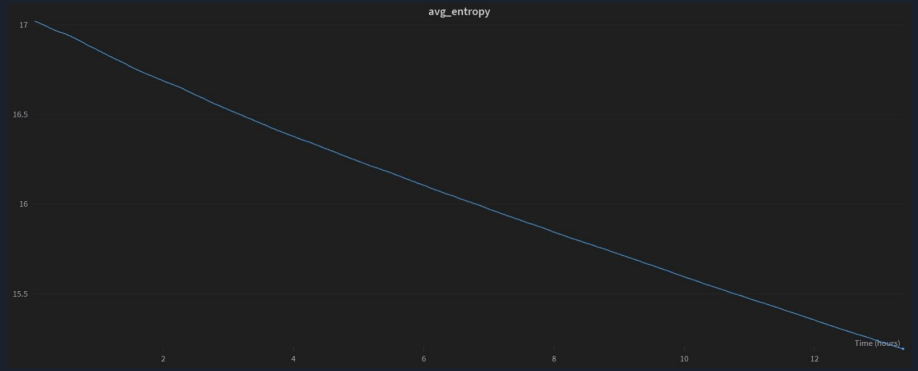
-506

Results Anybotics Anymal C



Results Anybotics Anymal C





Results Anybotics Anymal C

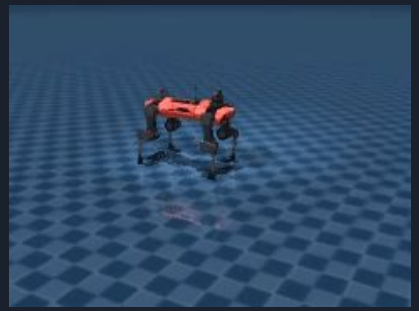
1300

2222

3347

4286

5213

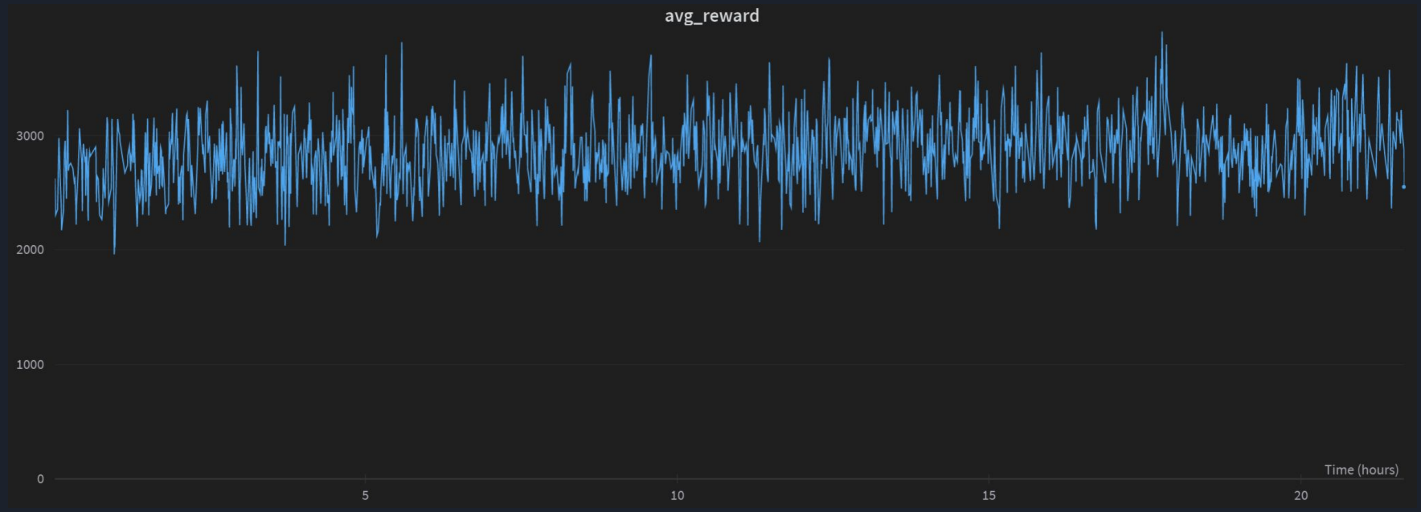


6134

7647

8325

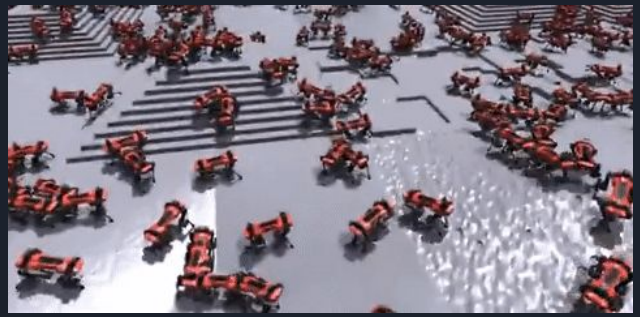
Results Anybotics Anymal C with a step



Conclusions

- PPO is a powerful algorithm that proved that with a small NN is capable of learning quite fast, solving these particular experiments in less than 24 hours training.
- Hyperparameter tuning is essential to converge on a solution but can take a lot of time.
- Transfer learning is possible when the agent and the environment are the same even when the NN is overfitted for a concrete task

Next steps to improve results:



Keep on working with hyper parameter tuning.

Training Multiple Actors and parallelizing with GPU.

Add more data to the state: last actions taken, collisions, terrain, etc.

Reward tuning, for example: penalizing energy consumption to optimize movements and make them smoother.

Change the entropy non-linearly or take the value for the covariance matrix from a NN.

Test with a bigger neural network.

Thanks for your attention