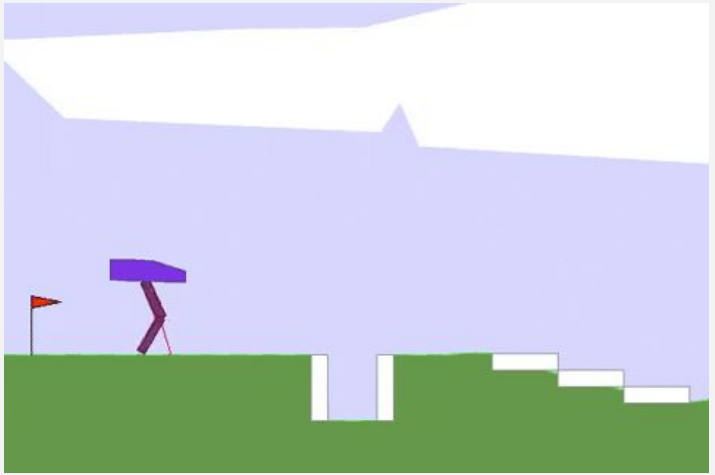
Bipedal Walker-v3

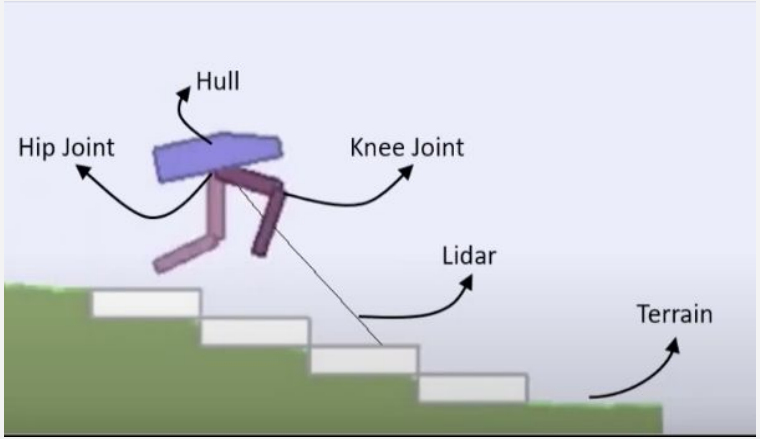
Work assembled by group N:
- Alejandro Gonçalves (up202205564)
- Francisca Mihalache (up202206022)
- João Sousa (up202205238)

❓How did we get here❓



Video from the TQC model on the 30.000.000th iteration

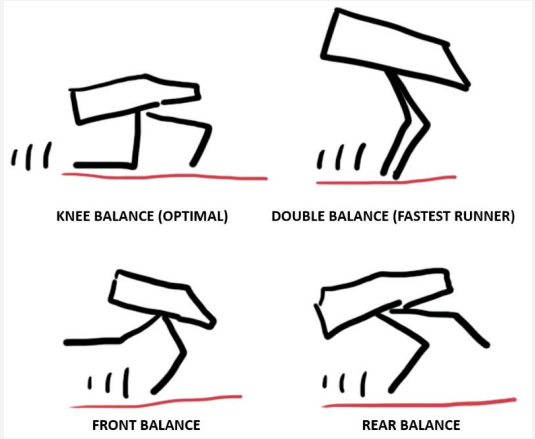
Environment & Agent:



Num	Observation	Min	Max	Mean
0	hull_angle	0	2*pi	0.5
1	hull_angularVelocity	-inf	+inf	-
2	vel_x	-1	+1	-
3	vel_y	-1	+1	-
4	hip_joint_1_angle	-inf	+inf	-
5	hip_joint_1_speed	-inf	+inf	-
6	knee_joint_1_angle	-inf	+inf	-
7	knee_joint_1_speed	-inf	+inf	-
8	leg_1_ground_contact_flag	0	1	-
9	hip_joint_2_angle	-inf	+inf	-
10	hip_joint_2_speed	-inf	+inf	-
11	knee_joint_2_angle	-inf	+inf	-
12	knee_joint_2_speed	-inf	+inf	-
13	leg_2_ground_contact_flag	0	1	-
14-23	10 lidar readings	-inf	+inf	-

Note that agent doesn't know anything about the way where he is running.

Walking strategies:



Positive Reward:

- Proportional to the distance covered (up to **300+ points**).

Negative Reward:

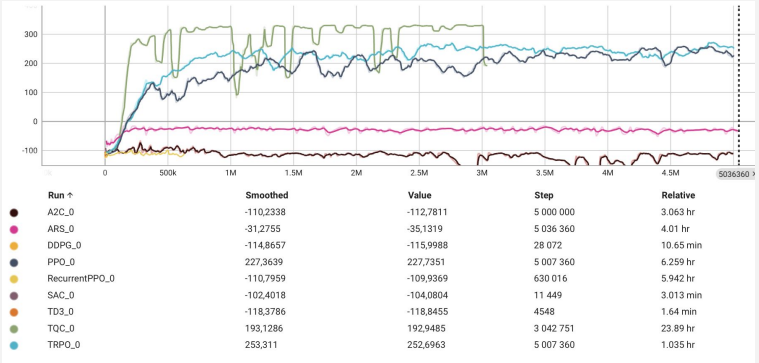
- -100 points for falling.
- Small penalty for torque usage to promote smooth, efficient movement.

Num	Name	Min	Max
0	Hip_1 (Torque / Velocity)	-1	+1
1	Knee_1 (Torque / Velocity)	-1	+1
2	Hip_2 (Torque / Velocity)	-1	+1
3	Knee_2 (Torque / Velocity)	-1	+1

Stable Baseline 3 -RL Algorithms

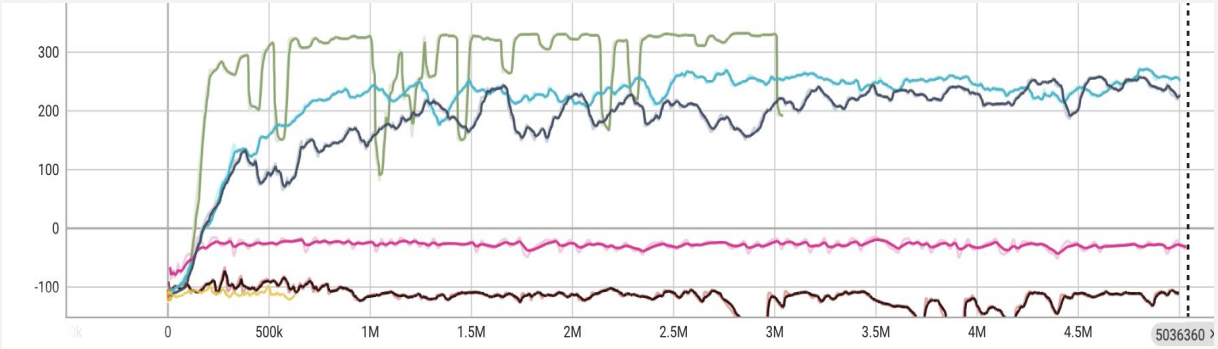
Name	Box	Discrete	MultiDiscrete	MultiBinary	Multi Processing
ARS [1]	✓	✓	✗	✗	✓
A2C	✓	✓	✓	✓	✓
CrossQ [1]	✓	✗	✗	✗	✓
DDPG	✓	✗	✗	✗	✓
DQN	✗	✓	✗	✗	✓
HER	✓	✓	✗	✗	✓
PPO	✓	✓	✓	✓	✓
QR-DQN [1]	✗	✓	✗	✗	✓
RecurrentPPO [1]	✓	✓	✓	✓	✓
SAC	✓	✗	✗	✗	✓
TD3	✓	✗	✗	✗	✓
TQC [1]	✓	✗	✗	✗	✓
TRPO [1]	✓	✓	✓	✓	✓
Maskable PPO [1]	✗	✓	✓	✓	✓

- ARS
- A2C
- HER
- PPO
- Recurrent PPO
- SAC
- TD3
- TQC
- TRPO



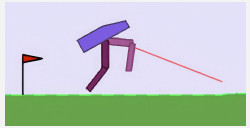
Run	Smoothed	Value	Step	Relative
A2C_0	-110,2338	-112,7811	5 000 000	3.063 hr
ARS_0	-31,2755	-35,1319	5 036 360	4.01 hr
DDPG_0	-114,8657	-115,9988	28 072	10.65 min
PPO_0	227,3639	227,7351	5 007 360	6.259 hr
RecurrentPPO_0	-110,7959	-109,9369	630 016	5.942 hr
SAC_0	-102,4018	-104,0804	11 449	3.013 min
TD3_0	-118,3786	-118,8455	4548	1.64 min
TQC_0	193,1286	192,9485	3 042 751	23.89 hr
TRPO_0	253,311	252,6963	5 007 360	1.035 hr

Model´s performance on the original base environment:



Run ↑	Smoothed	Value	Step	Relative
A2C_0	-110,2338	-112,7811	5 000 000	3.063 hr
ARS_0	-31,2755	-35,1319	5 036 360	4.01 hr
DDPG_0	-114,8657	-115,9988	28 072	10.65 min
PPO_0	227,3639	227,7351	5 007 360	6.259 hr
RecurrentPPO_0	-110,7959	-109,9369	630 016	5.942 hr
SAC_0	-102,4018	-104,0804	11 449	3.013 min
TD3_0	-118,3786	-118,8455	4548	1.64 min
TQC_0	193,1286	192,9485	3 042 751	23.89 hr
TRPO_0	253,311	252,6963	5 007 360	1.035 hr

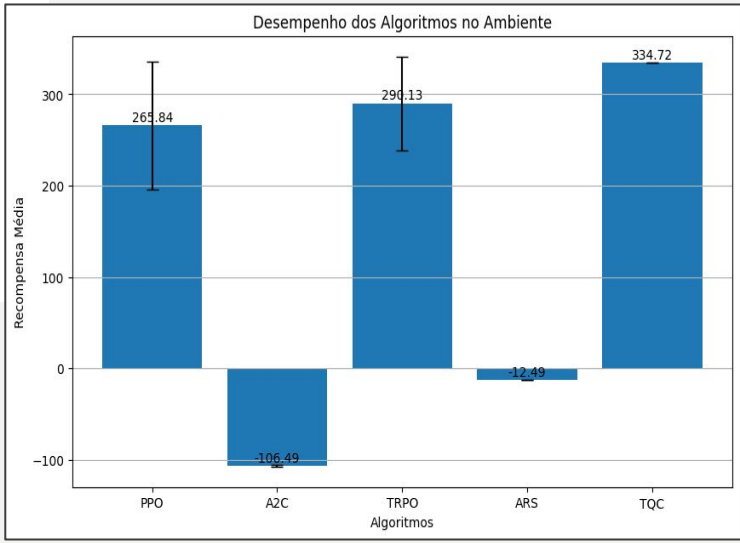
Average reward per episode during the training of all possible models on the original environment



There are 3 models that clearly outperform all the others:

- **PPO**
- **TQC**
- **TRPO**

Therefore, we will focus on these.



Our 3 chosen models:

PPO

Improves policies by optimizing a clipped surrogate objective function. It balances exploration and exploitation by using a trust region to ensure stable updates. PPO is widely used due to its simplicity, efficiency, and robustness.

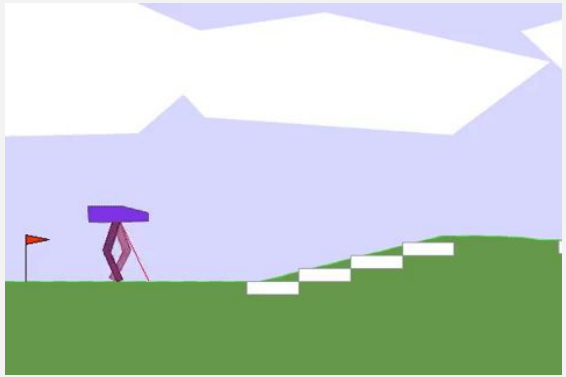
TQC

Variant of the TD3 algorithm designed for continuous action spaces. It uses multiple critics and employs a truncation mechanism to focus on the most reliable Q-value estimates, improving stability and reducing overestimation bias.

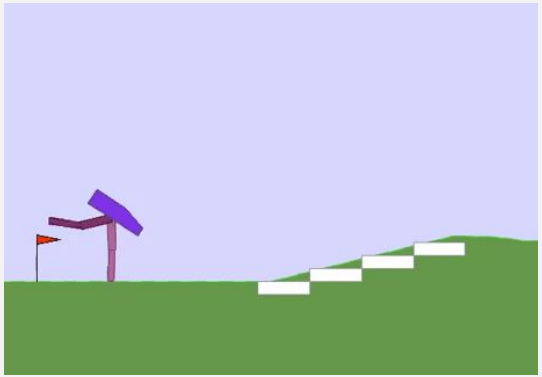
TRPO

TRPO is a policy optimization algorithm that ensures safe updates by constraining the step size within a trust region. It uses the Kullback-Leibler (KL) divergence as a constraint to maintain stability and prevent drastic changes in policy.

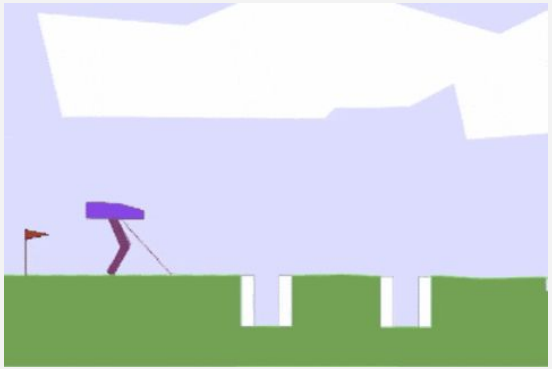
The beginning...



PPO 1st iteration

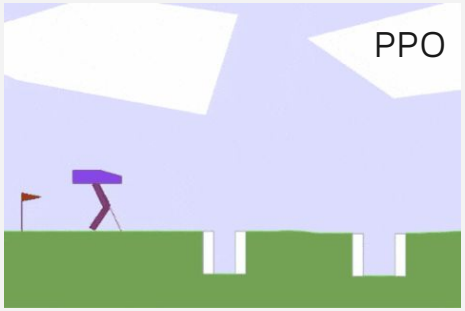


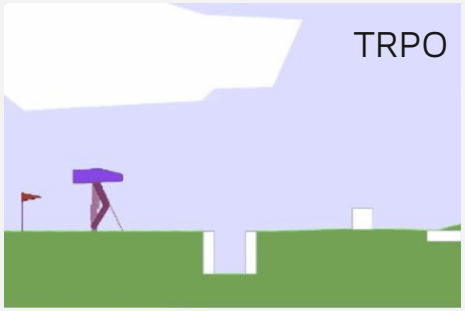
TQC 1st iteration



TRPO 1st iteration

Hard environment:



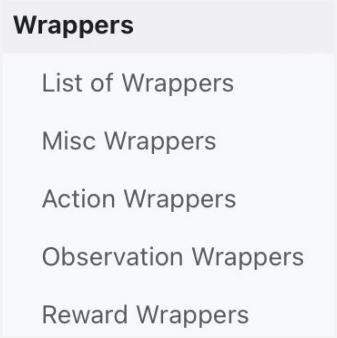



```
env_id = "BipedalWalkerHardcore-v3"
```

Videos from iteration 10 million

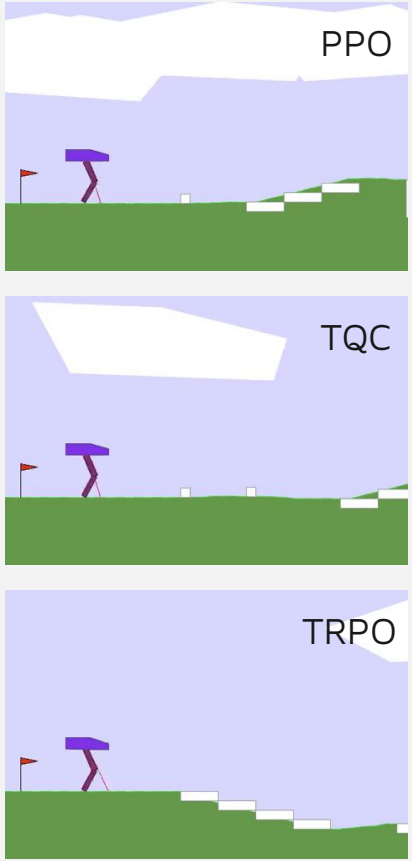
Wrapped environments:

“”Wrappers are a convenient way to modify an existing environment without having to alter the underlying code directly.””
– Gymnasium Documentation -



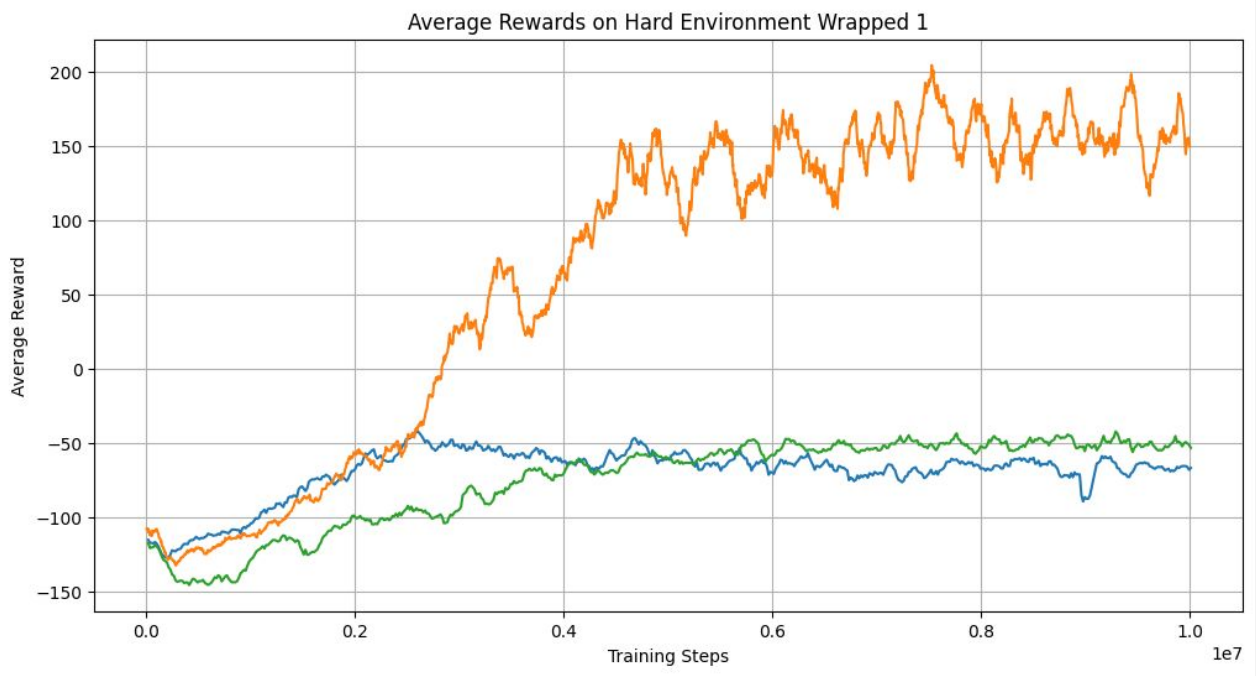
They are used to adjust observations, rewards, or other aspects of the environment to improve training efficiency and adaptability.

Wrapped environment 1:

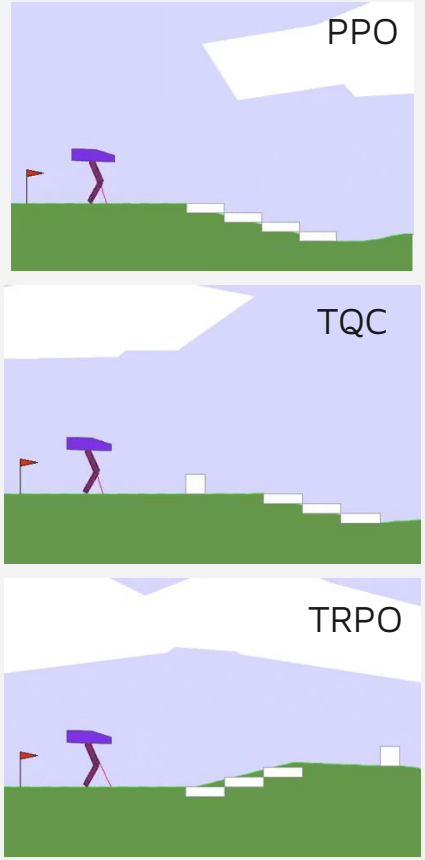


Videos from iteration 10 million

NormalizeObservation: Normalizes observations to stabilize training (mean=0 & standard deviation=1
FrameStackObservation: Stacks the last 4 observations to provide temporal context.
RecordEpisodeStatistics: Tracks rewards, durations, and performance metrics.
TimeLimit: Limits each episode to 2000 steps to avoid infinite loops.

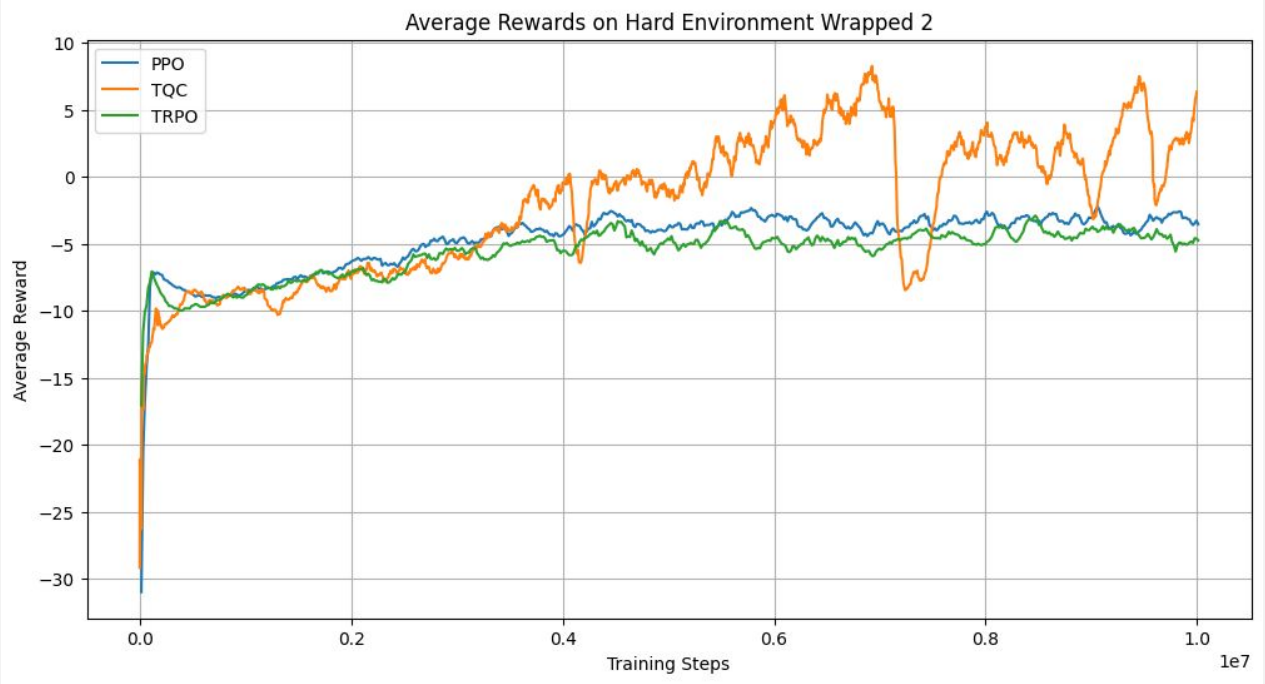


Wrapped environment 2:



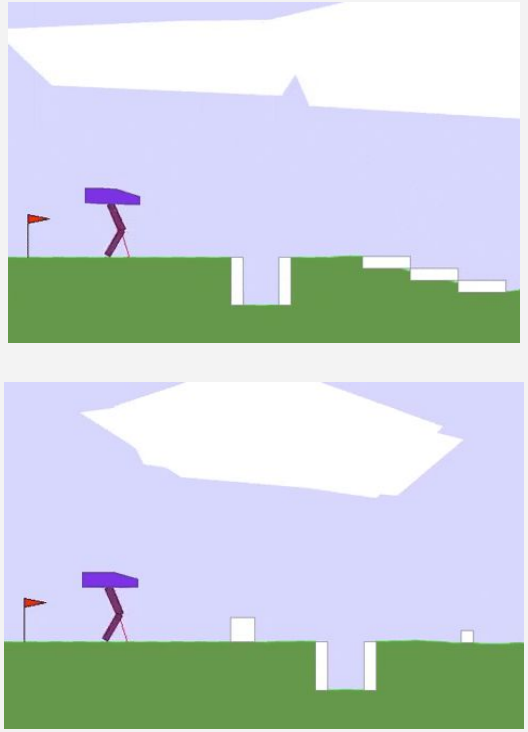
Videos from iteration 10 million

NormalizeReward: Scales rewards to stabilize learning.
TransformReward: Adjusts rewards by penalizing angles and rewarding progress.
Monitor: Logs rewards and episode lengths detailed



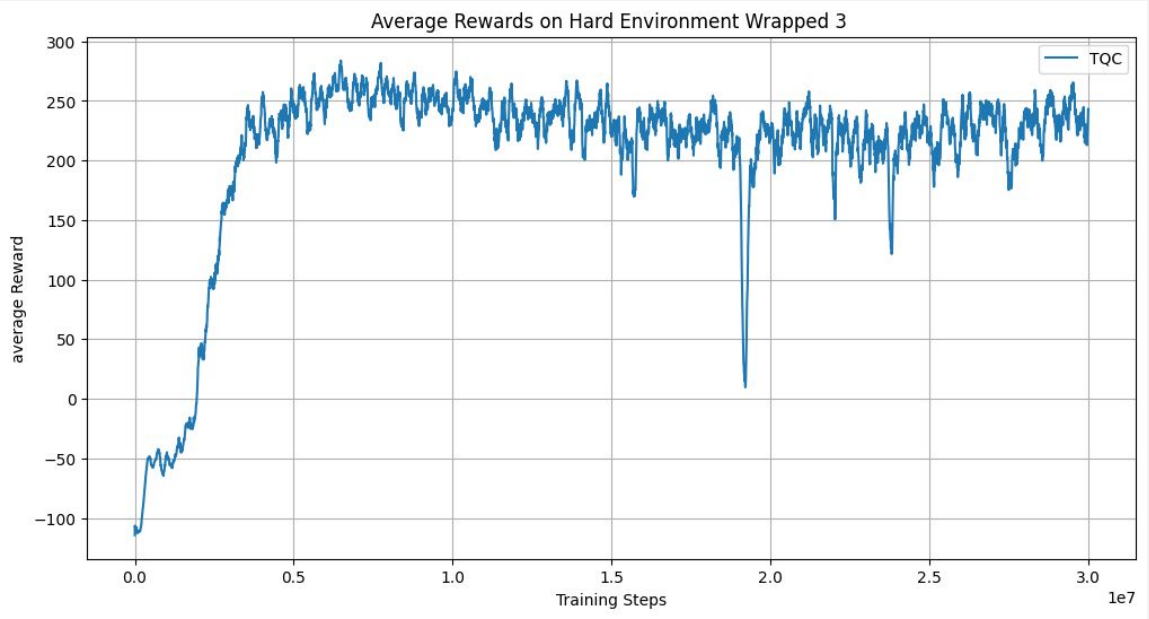
Unfortunately, this model was not our best choice. By modifying the rewards in this way, we lose a consistent basis for comparison with other algorithms. However, we were able to observe that its performance is worse than the other models.

Wrapped environment 3:

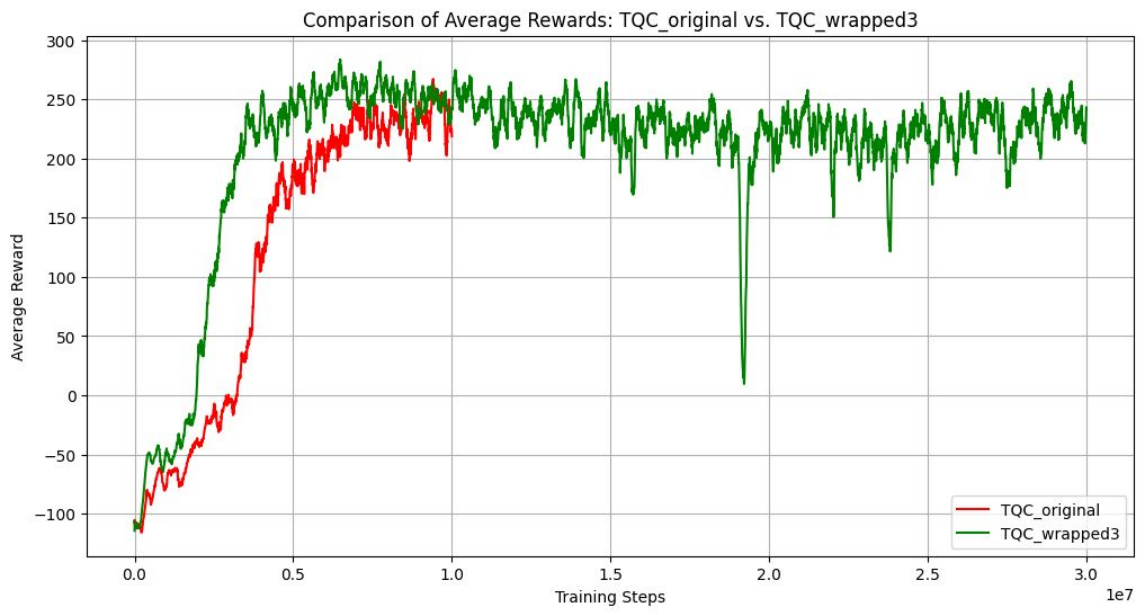


Videos from iterations 30 million

RescaleAction: Ensures that the actions taken by the agent are scaled between -1.0 and 1.0, keeping them within valid bounds.
InfoCaptureWrapper: Captures and stores the info dictionary from each step for use in reward adjustments.
CustomRewardWrapper: Modifies the rewards by introducing penalties for large angles and excessive leg openings, aiming to encourage stable and coordinated movement.
TimeAwareObservation: Adds normalized time as part of the observations, enabling the agent to account for the passage of time during training.



Comparison between TQC original and TQC wrapped 3



Although the **TQC_original** model was not trained for the full 30 million iterations as the **TQC_wrapped3** model, the results clearly indicate a superior performance of the **TQC_wrapped3**.

- The **average reward** for **TQC_wrapped3** consistently outperforms that of **TQC_original** after the initial training phase.
- The **TQC_wrapped3** shows a faster convergence, achieving higher rewards earlier in the training process compared to **TQC_original**.

Conclusions:

Why TQC Outperformed PPO and TRPO ?

Feature	TQC	PPO / TRPO
Learning Type	Off-policy: Utilizes stored data from a replay buffer for sample-efficient learning.	On-policy: Only learns from data collected by the current policy, less efficient.
Designed For	Complex continuous control environments.	Originally designed for discrete environments, adapted for continuous control.
Handling Outliers	Reduces impact of extreme values by ignoring higher quantiles (positive outliers).	No specific mechanism for handling outliers, making learning less stable.
Sample Efficiency	Requires fewer samples to learn effective policies.	Requires constant new sampling, increasing computational costs.
Exploration vs Exploitation	Balances exploration and exploitation effectively via truncated quantile learning.	Prone to getting stuck in local minima due to gradient-based optimization.
Adaptability to Custom Rewards	More capable of benefiting from custom reward systems due to its flexible Q-value critics.	Less flexible in leveraging custom reward structures.

Future work:

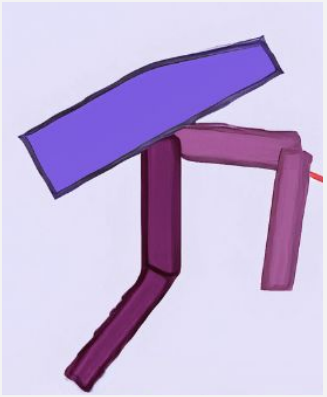
- Enhance the complexity of the environment by adding flying objects or dynamic obstacles.
- Conduct experiments with extended training iterations
- Perform an in-depth optimization of hyperparameters for TQC, PPO, and TRPO.

Meet our team :)

João Sousa

Francisca Mihalache

Alejandro Gonçalves



Bipedal Walker

We are grateful for your time and interest!
Any questions? We’re here to share more ideas!