Solving OpenAI Challenge: AntWalker using Off-Policy Reinforcement Learning Methods

Rhea Sharma
Harshita Maddi

BE BOUNDLESS

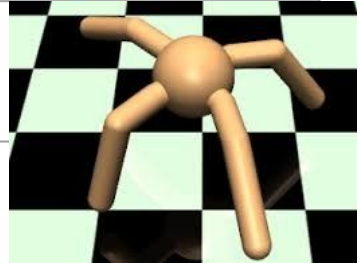
Project Motivation

The motivation behind choosing this project was to implement reinforcement learning solutions to understand its concepts even better. We were interested in off-policy learning and intrigued with simulations that wanted to understand multi-jointed motions. We used, MuJoCo, short for "Multi-Joint Dynamics with Contact," which is a physics engine very commonly used in animations and games. Historically, they have been used to evaluate the performance of newly proposed reinforcement learning algorithms.

ANT MUJOCO Environment

One of the MuJoCo, environments is the ant, which is a 3D robot consisting of one torso (free rotational body) with four legs attached to it where each leg has two links. The goal is to coordinate the four legs to move in the forward (right) direction by applying torques on the eight hinges connecting the two links of each leg and the torso (nine parts and eight hinges).

Actions : Torques applied at the hinge joints **Observation Space:** Positional values of different body parts of the ant, followed by the velocities of those individual parts	**Rewards:** • Healthy_reward: Every timestep that the ant is healthy • Forward_reward: A reward of moving forward • Ctrl_cost: Negative reward for penalising the ant if it's actions are too large • Contact_cost: Negative reward for penalising the ant if the external contact force is too large ***reward = healthy_reward + forward_reward - ctrl_cost - contact_cost***
Starting State : An ant that is standing with it's orientation facing forward	**Episode termination :** 1. Any of the state space values are not finite 2. The z-coordinate of the torso is not in healthy range



Sample Environment Video

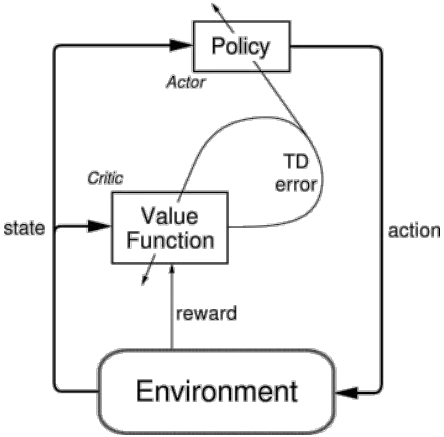
Without enough training, the ant finds it difficult to move around and has no intuition of new actions to take.

SAC

Soft Actor-Critic (SAC) is a popular reinforcement learning algorithm that can be used to learn policies for continuous control tasks. SAC is a value-based actor-critic algorithm that uses three neural networks: an actor network, a critic network, and a temperature parameter network.

The actor network takes the current state as input and outputs a probability distribution over the actions the agent can take. The critic network takes the current state and action as input and estimates the expected cumulative reward. The temperature parameter network is used to adjust the entropy of the policy distribution to balance exploration and exploitation.

One of the key benefits of SAC is that it can learn policies for tasks with continuous action spaces, which can be difficult for other RL algorithms. SAC is also known for its sample efficiency, meaning that it can learn from fewer interactions with the environment than other algorithms.



http://incompleteideas.net/book/first/ebook/node66.html

DDPG Limitations + introduce TD3

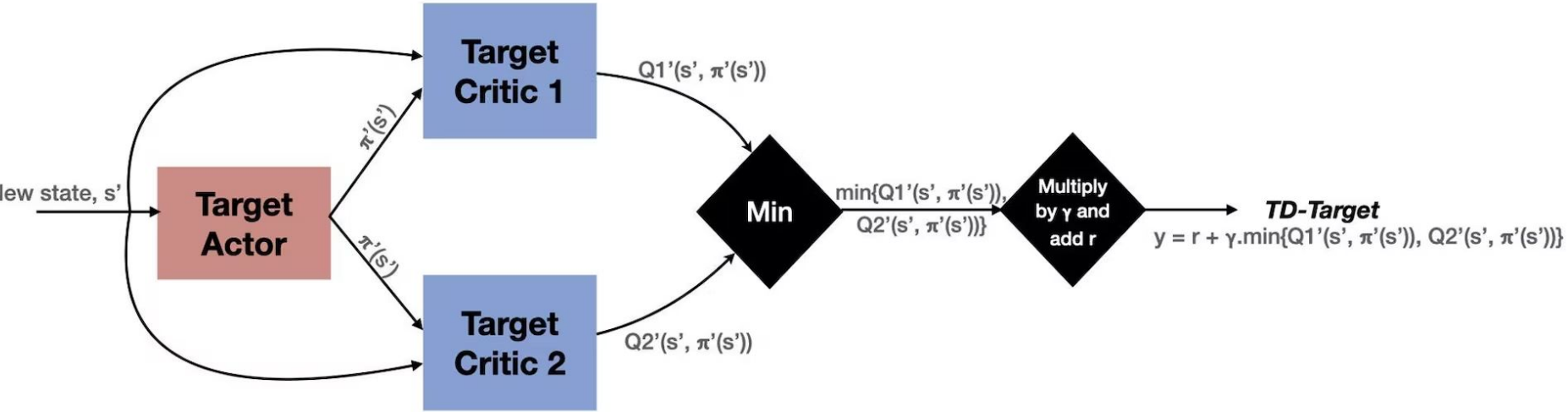


Image from:
https://saashanair.com/blog/blog-posts/twin-delayed-ddpg-td3-how-does-the-algorithm-work

Experiments

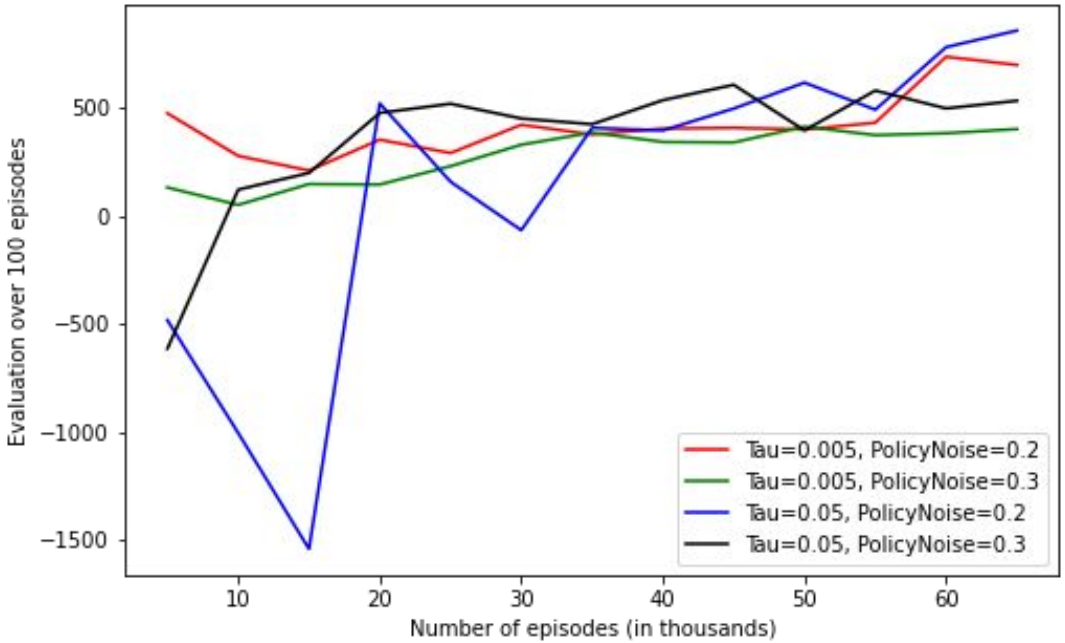
We evaluated four different TD3 models with the following configurations and compared the evaluation of the best policies in each case.

OBSERVATION = 10000
EXPLORATION = 70000
BATCH_SIZE = 100
GAMMA = 0.99
NOISE_CLIP = 0.5
EXPLORE_NOISE = 0.1
POLICY_FREQUENCY = 2
EVAL_FREQUENCY = 5000
REWARD_THRESH = 8000

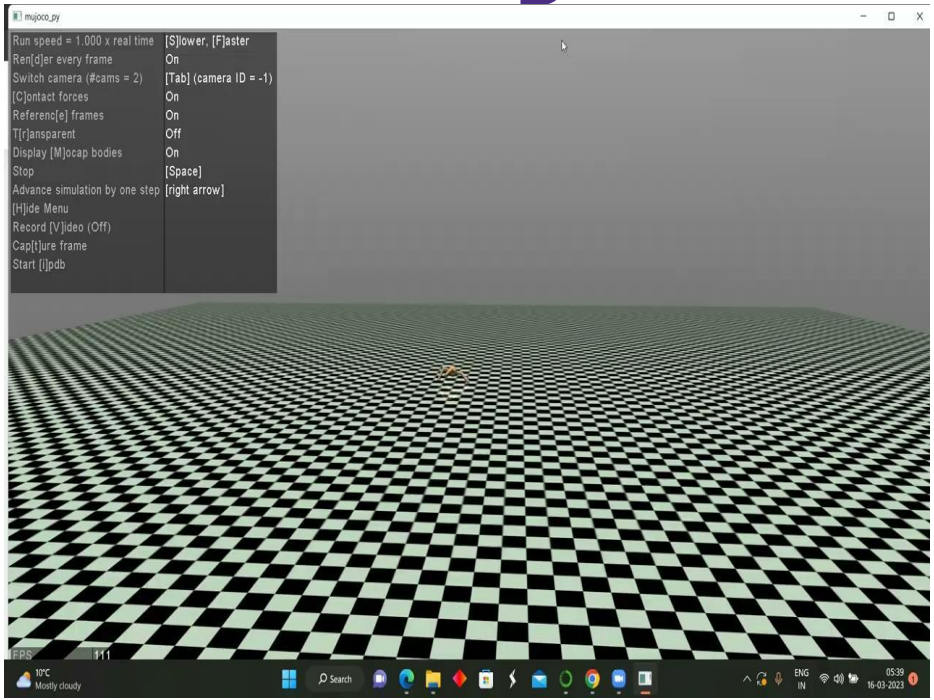
	POLICY_NOISE = 0.2	POLICY_NOISE = 0.3
TAU = 0.05	**855.17**	530.89
TAU = 0.005	696.26	400.042

Evaluation reward over 100 episodes

Comparison of TD3 models evaluation



Ant Walker Demo using TD3



UNIVERSITY of WASHINGTON

Further Research

As part of our further research, we would like to study algorithms such as TRPO, PPO and A3C for continuous control .

These are on-policy algorithms. Training is faster in A3C but the convergence is better is in PPO and hence we would like to explore these RL algorithms.