Universität Stuttgart

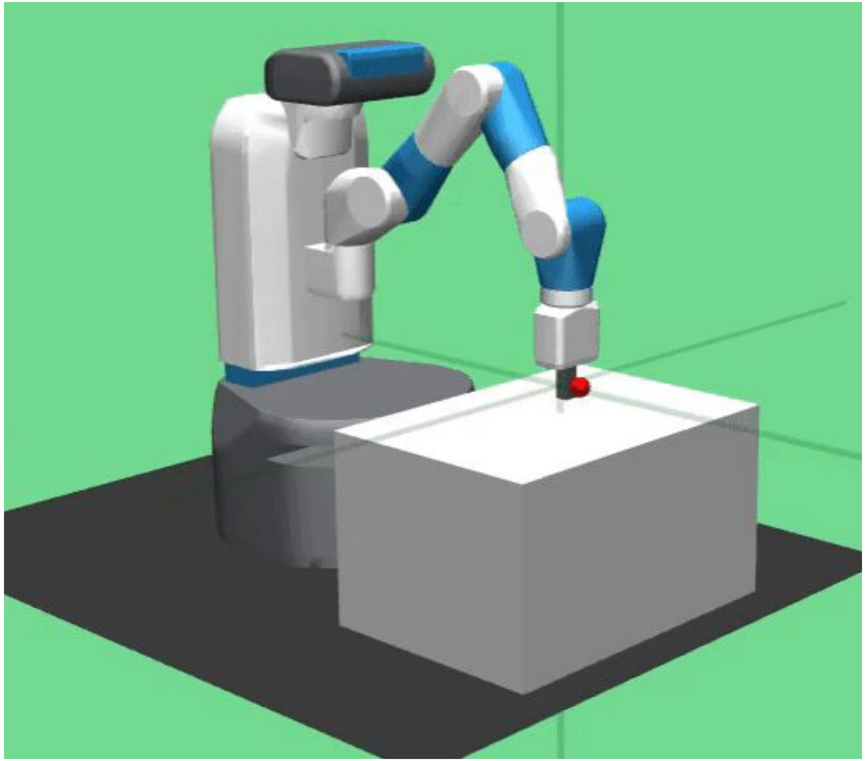
Intelligent Object Sorting using Deep Reinforcement Learning Robot & Computer Vision

Robotics Lab, Winter Term 2021-22

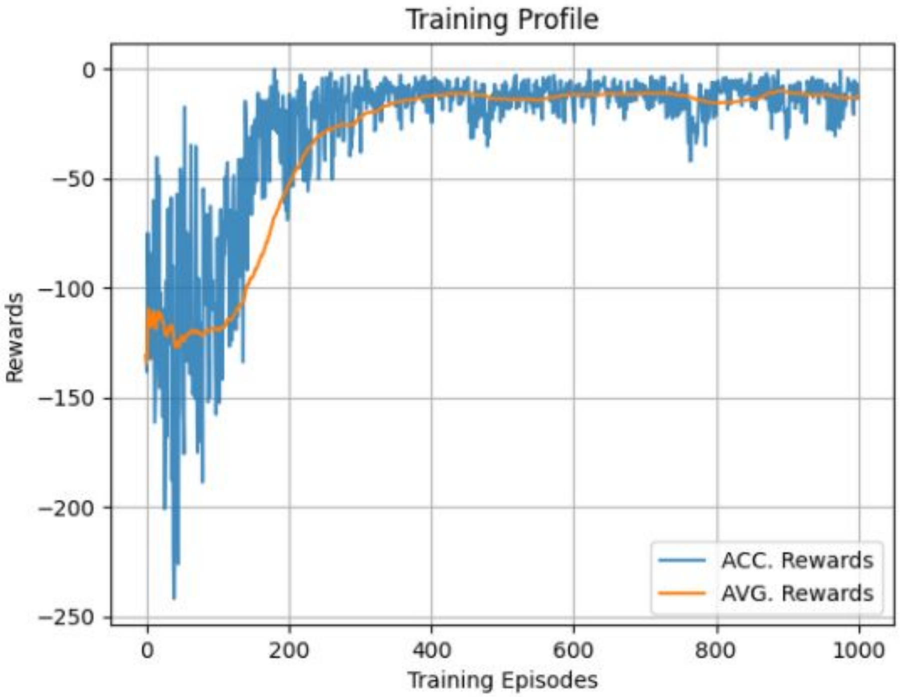
Kanishk Navale / 3437531

Olga Klimashevska / 3525388

Proof of Concept

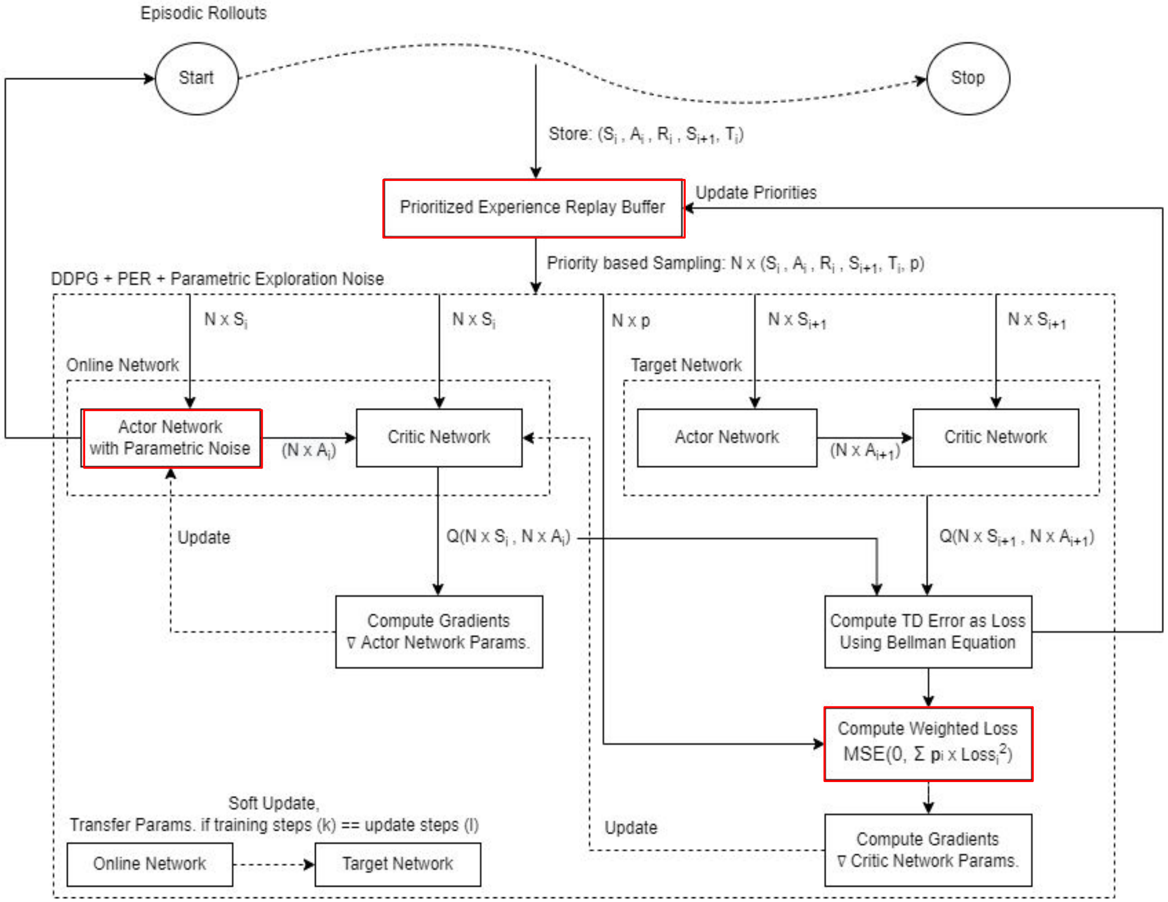


Benchmarking DDPG, TD3 & PPO on openai-gym “FetchReach-v1”



Training Profile of the DDPG Agent

Engineering a better DDPG Algorithm

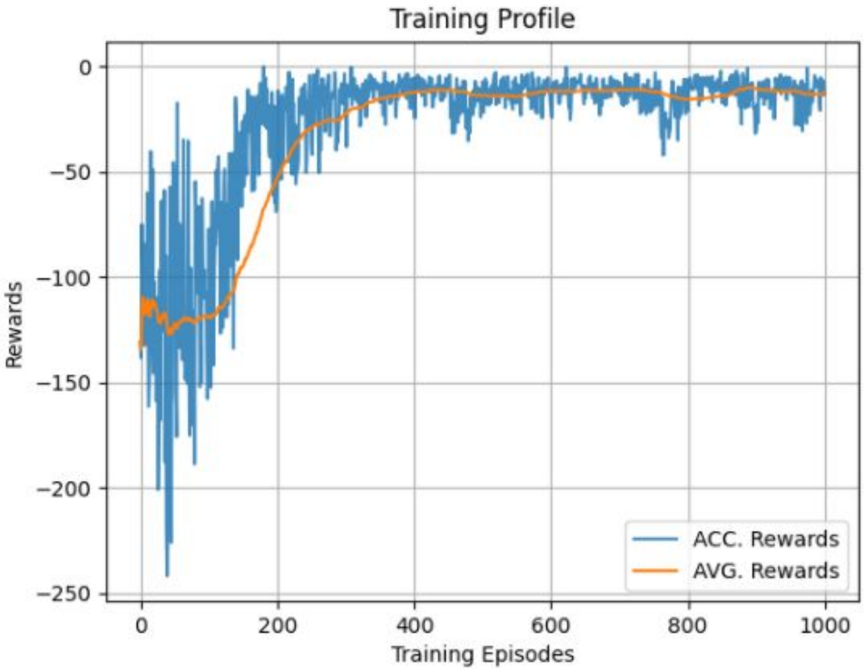

Episodic Rollouts
Start
Stop
Store: (S_i , A_i , R_i , S_i+1, T_i)
Prioritized Experience Replay Buffer
Update Priorities
Priority based Sampling: N x (S_i , A_i , R_i , S_i+1, T_i, p)
DDPG + PER + Parametric Exploration Noise
N x S_i
N x S_i
N x p
N x S_i+1
N x S_i+1
Online Network
Target Network
Actor Network with Parametric Noise
(N x A_i)
Critic Network
Actor Network
(N x A_i+1)
Critic Network
Update
Q(N x S_i , N x A_i)
Q(N x S_i+1 , N x A_i+1)
Compute Gradients ∇ Actor Network Params.
Compute TD Error as Loss Using Bellman Equation
Compute Weighted Loss MSE(0, Σ pi x Loss_i^2)
Soft Update, Transfer Params. if training steps (k) == update steps (l)
Online Network
Target Network
Update
Compute Gradients ∇ Critic Network Params.

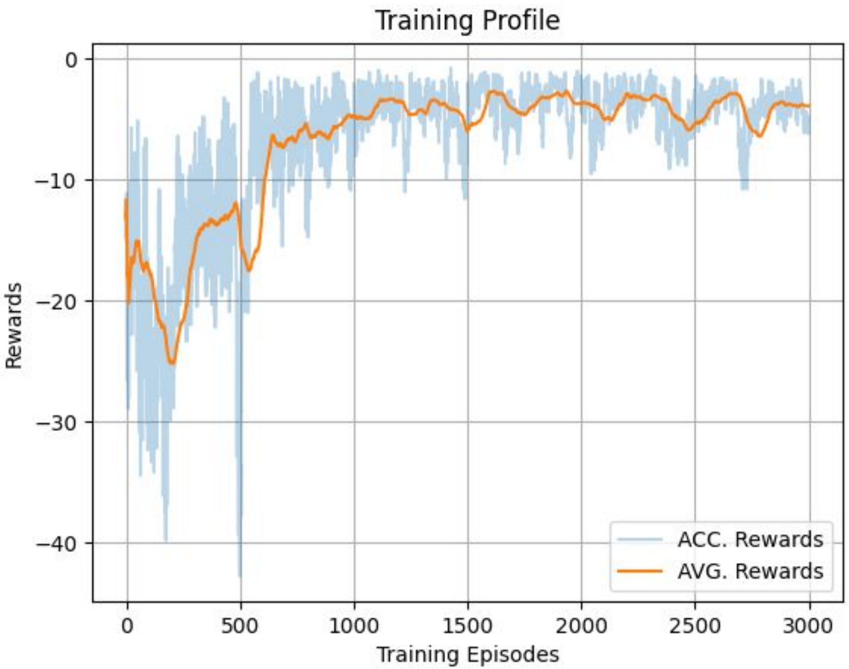
Building State & Action for Interaction

- For each play step in a game,
    - Build: state = Current Robot TCP(x, y, z) | Target Location P(x, y, z)
    - Compute: action = actor.choose_noisy_action(state)
    - Get: next_state, reward, done = env.step(action)
    - Reward = Negative Euclidean distance

- DDPG Agent is optimized to maximize the reward for each play step over the games.

Comparison of Vanilla DDPG & Our DDPG

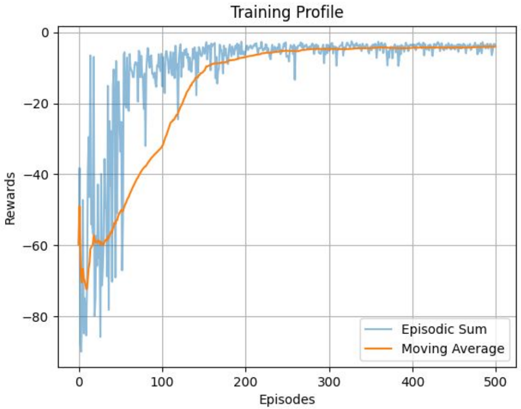


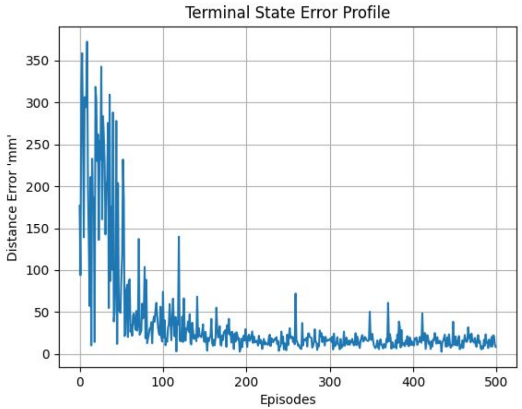


Result: The DDPG Agent is 5 times better (metric: training rewards) with PER & Parametric Exploration Noise.

Training DDPG Agent for Robot Motion

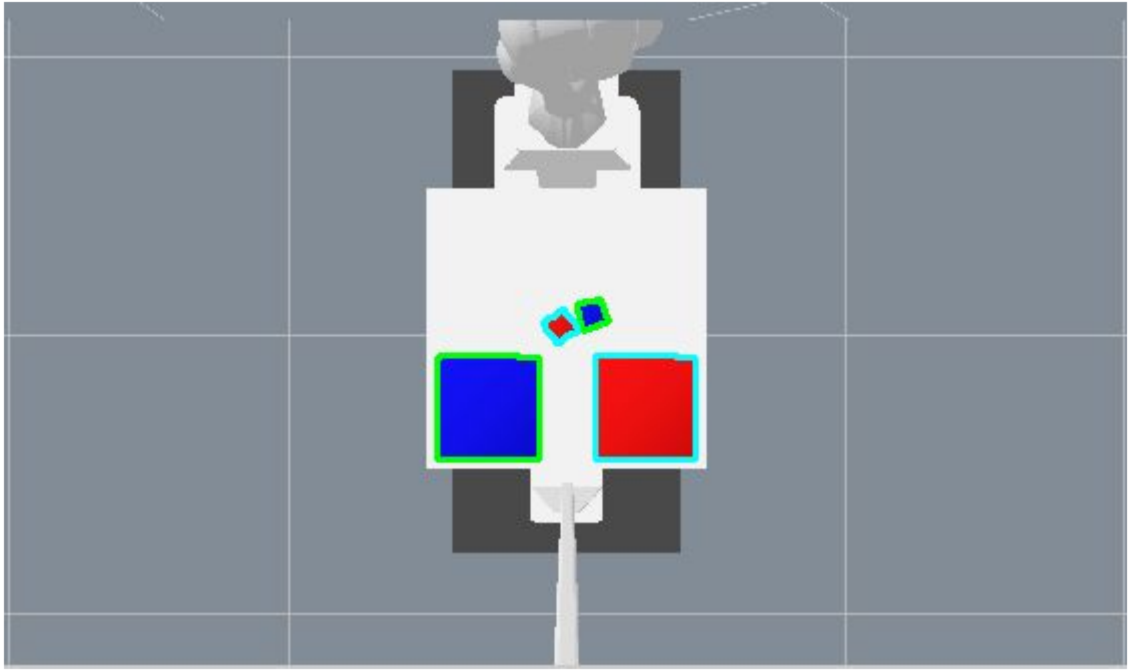
Training Episode: 0





1. SolidWorks is used to develop the meshes.
2. Meshes are exported & imported in ‘rai’ after processing .urdf files.
3. The ‘gym’ wrapper is used to create ‘reach_gym’ to train the robot.
4. It takes 1Hr. to train the robot for 500 episodes.
5. The robot reaches any point in the Cartesian space with error of ±5mm.
6. The training occupies 3GB of GPU.

Pose Estimation Pipeline



Object Tending