Balancing a Segway

End-Semester Presentation: 5-May-2025

RO3002: Fundamentals of Reinforcement Learning

By: Team Stay Up Segway (SUS)
Abhivarya Kumar (U20220006), Jia Bhargava (U20220046),
Tanay Srinivasa (U20220086), Vandita Lodha (U20220093).

Problem Description

Main Goal: Balance the segway hardware for as long as possible while not moving the robot from its initial position

Research Gap:
Implementation on Non-Ideal Hardware, Analysis of Control Effort

EoM:

$$F = m_c\ddot{x} + m_P(\ddot{x} + d\dot{\theta}^2 \sin\theta - d\ddot{\theta}cos\theta) + b\dot{x}$$

$$0 = \ddot{\theta}(I + m_p d) - m_p d(\ddot{x}\cos\theta + g\sin\theta)$$

Note: The dynamics, hardware, and baseline comparisons were initially completed as a part of the RO3003: Control Autonomy, Planning and Navigation course.

Literature Review

Mishra and Arora (2022) [1]

Applied DQN and Double DQN to CartPole.
Replaced squared loss with Huber loss for faster, more stable training.
Double DQN reduced Q-value overestimation: quicker and more accurate convergence.
Highlights: Stable loss functions and reducing overestimation to boost DQN performance

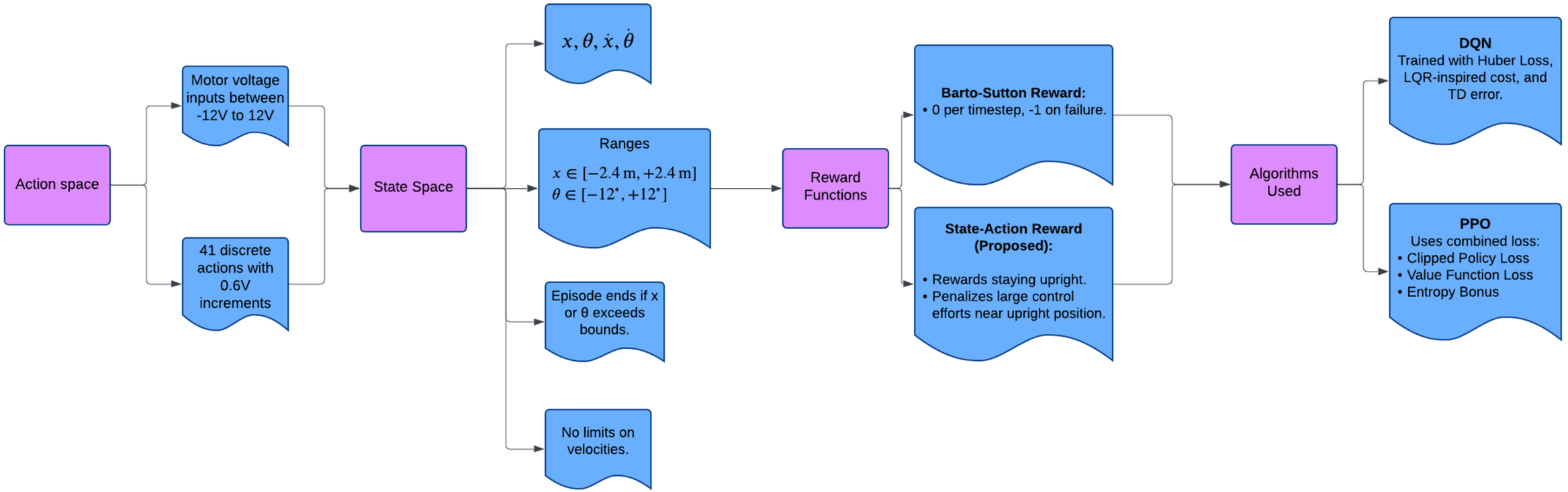
Rio, Jimenez, Serrano [2]

Compared PPO and A3C on CartPole.
PPO showed superior training stability due to controlled policy updates
Required longer execution times (~452s), and achieved a 10.5% CartPole success rate.

Jo and Kim (2024) [3]

Compared DQN, PPO, and A2C on CartPole.
DQN outperformed PPO and A2C: faster convergence, higher rewards, and lower variance.
DQN’s success was due to experience replay and exploration-exploitation management

RL Formulation



Methodology

Agents: DQN, PPO

Simulation Environment:
Custom Gymnasium CartPole Environement

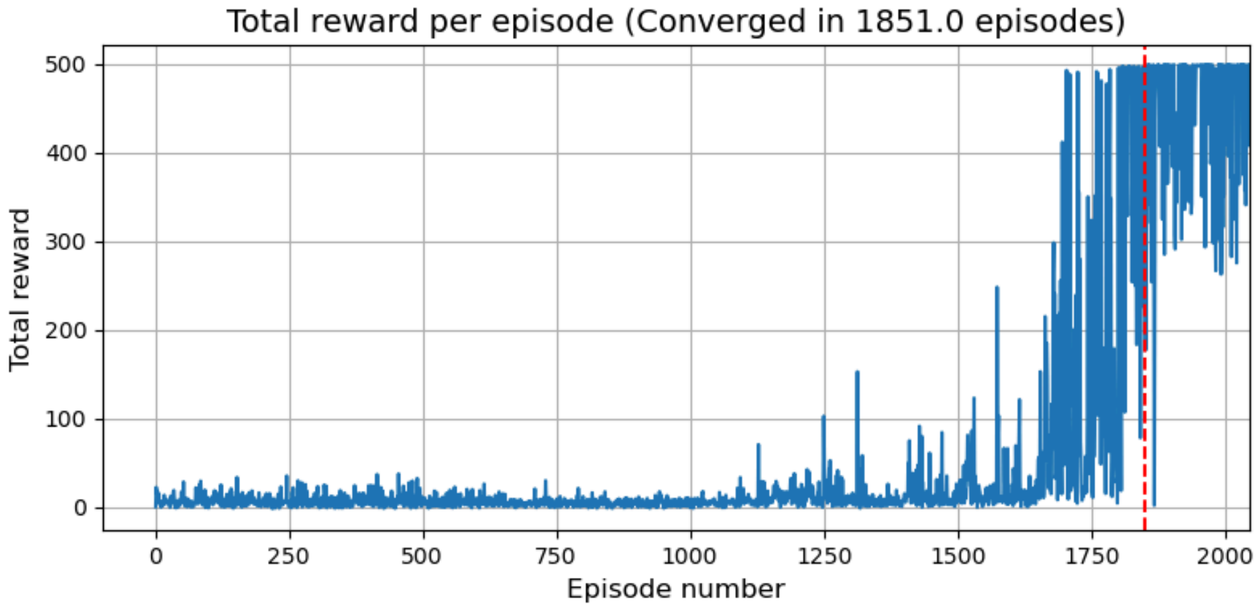
Reward Function:

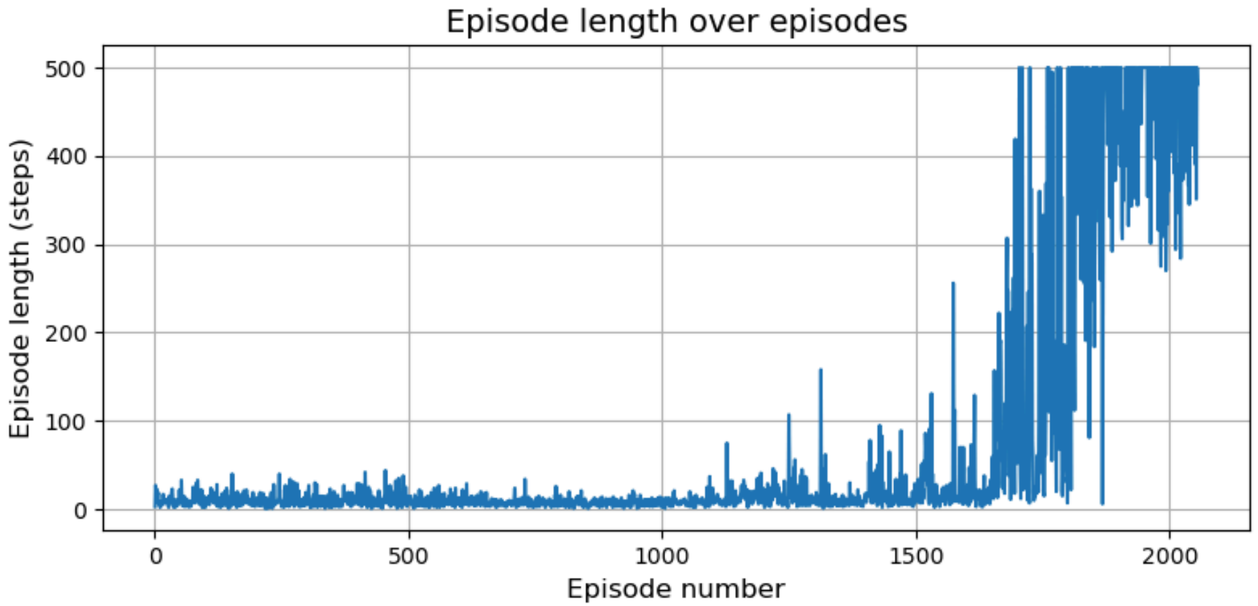
$$R_t = cos^2(\theta) - (1 - cos(\theta)) \times k \times (action - 20)^2$$

Experimental Setup:
Hardware Components: 500 RPM DC Motor w/ Encoders, Raspberry Pi 3B+, MPU6050, Cytron MDD10A
Compensations: Motor Deadband, IMU Sensor Fusion, Encoder and Motor LPF
Training Strategy: Sim → Hardware Fine-Tuning → Hardware Deployment

Results:

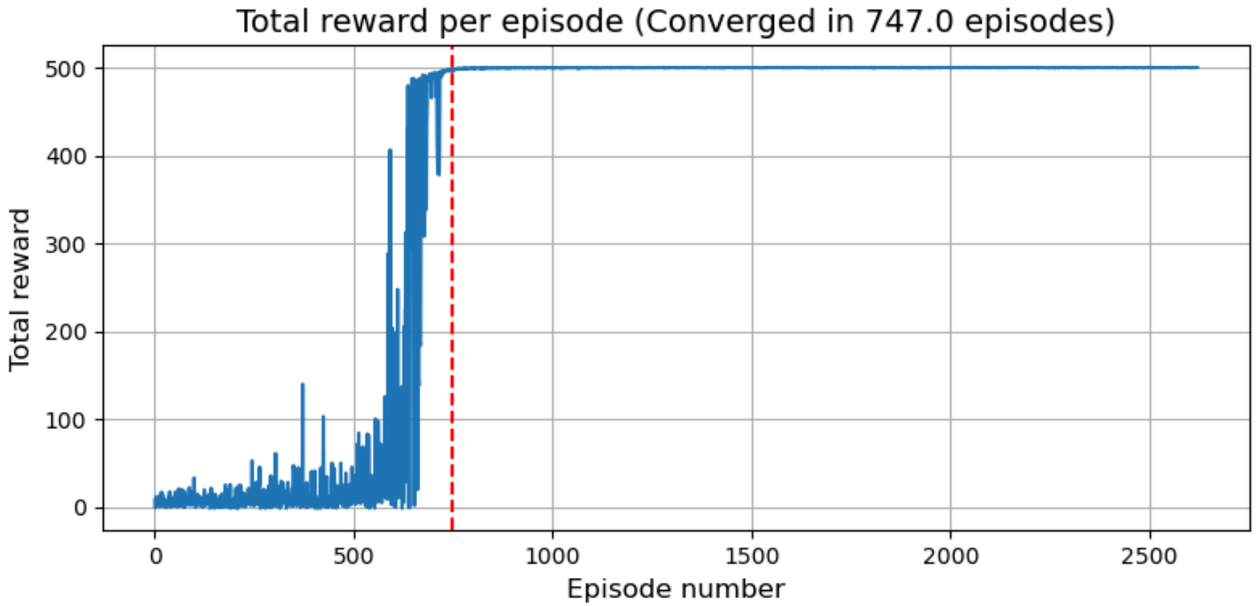
Training in Simulation: DQN

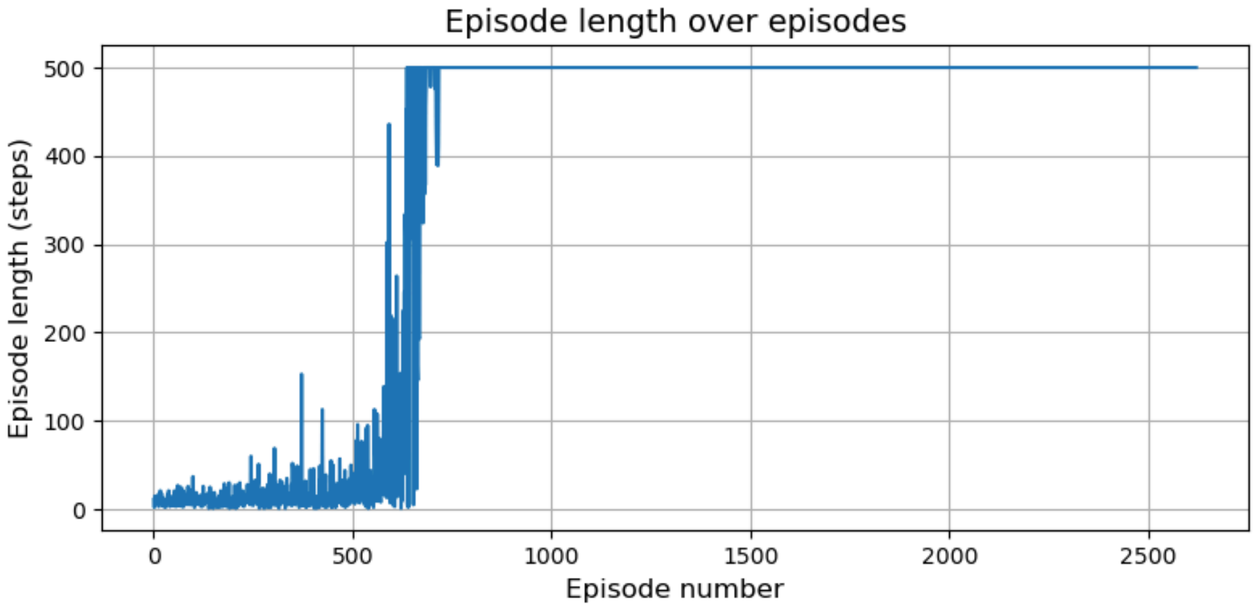




Results:

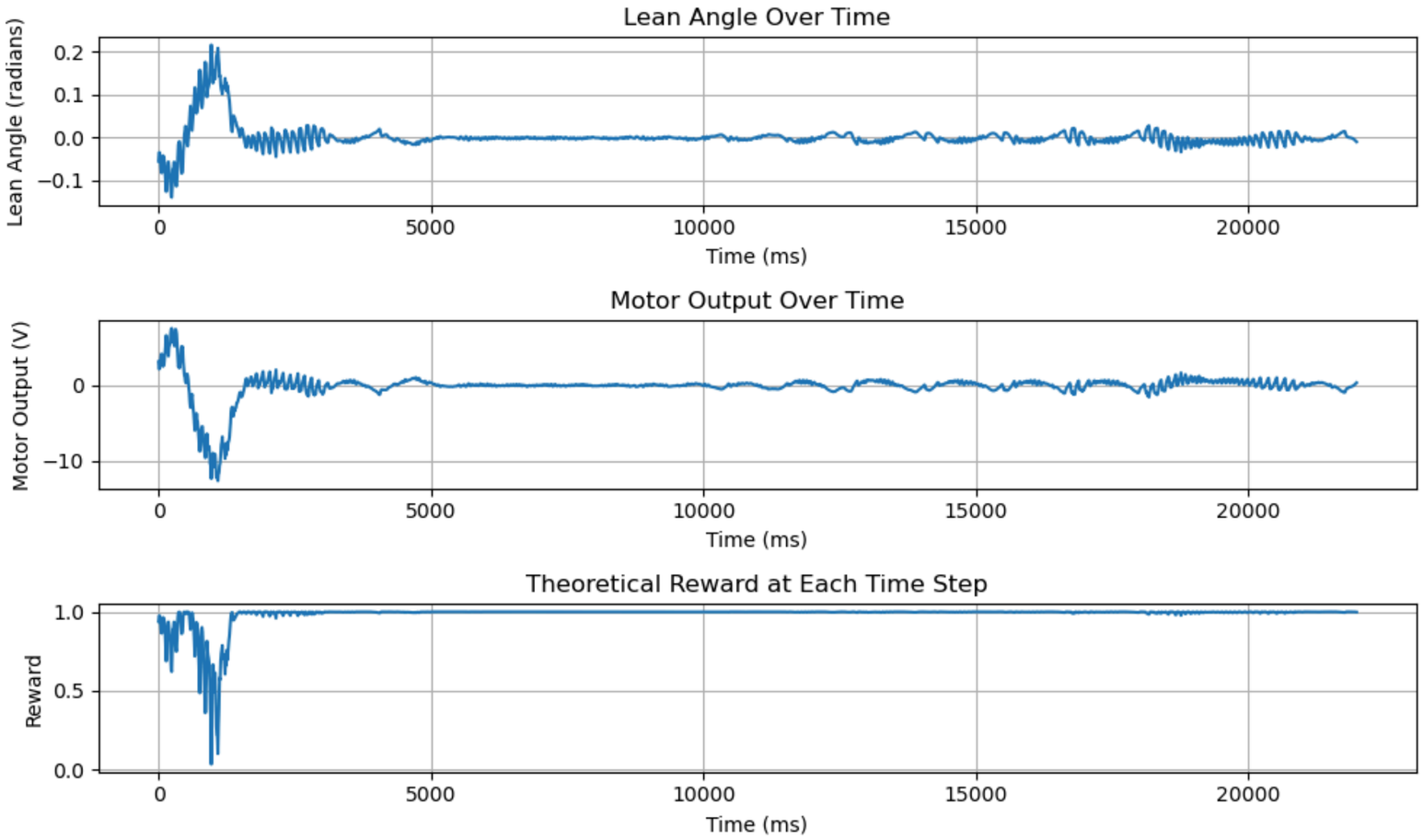
Training in Simulation: PPO





Results:

Hardware Deployment: PID Baseline

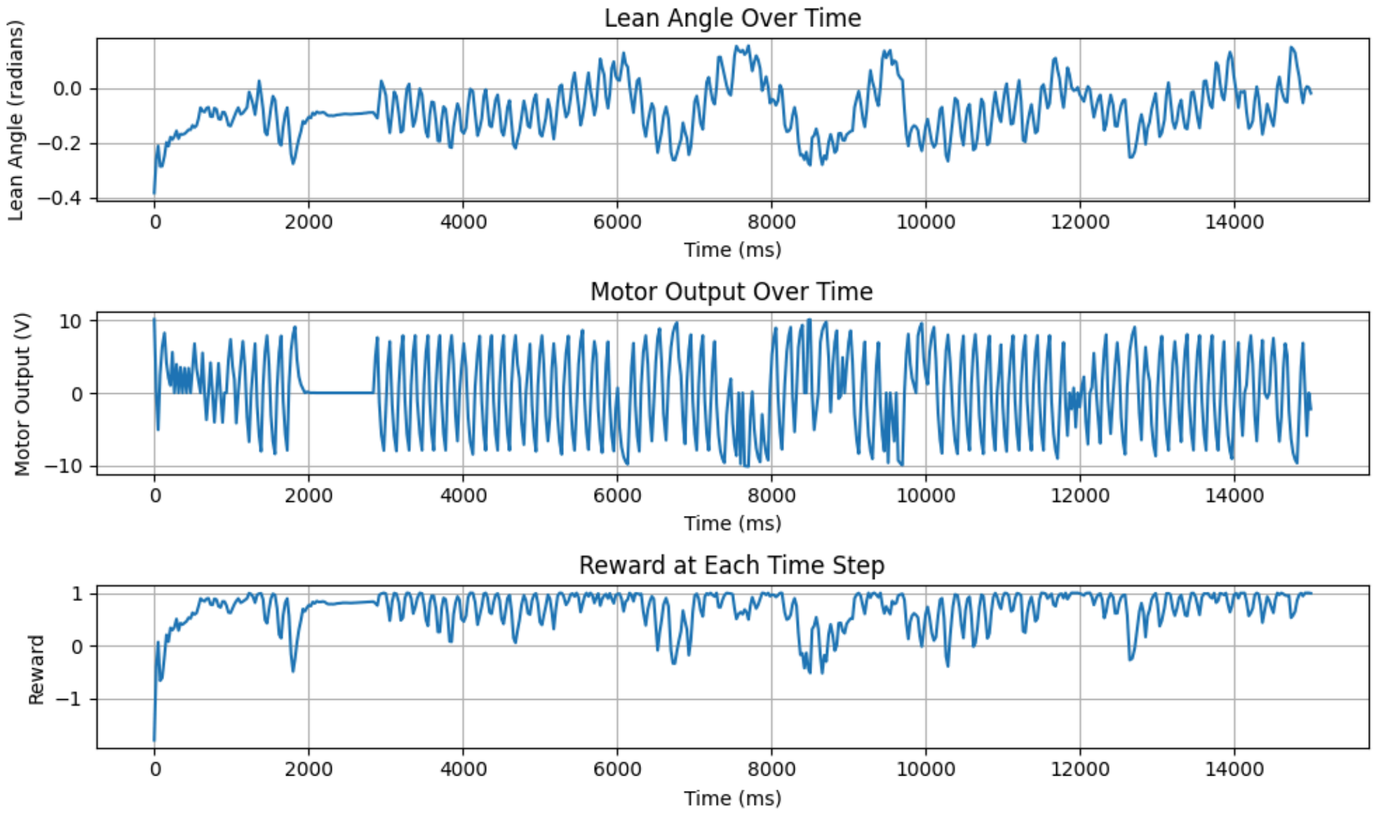


Results:

Hardware Deployment: PID Baseline

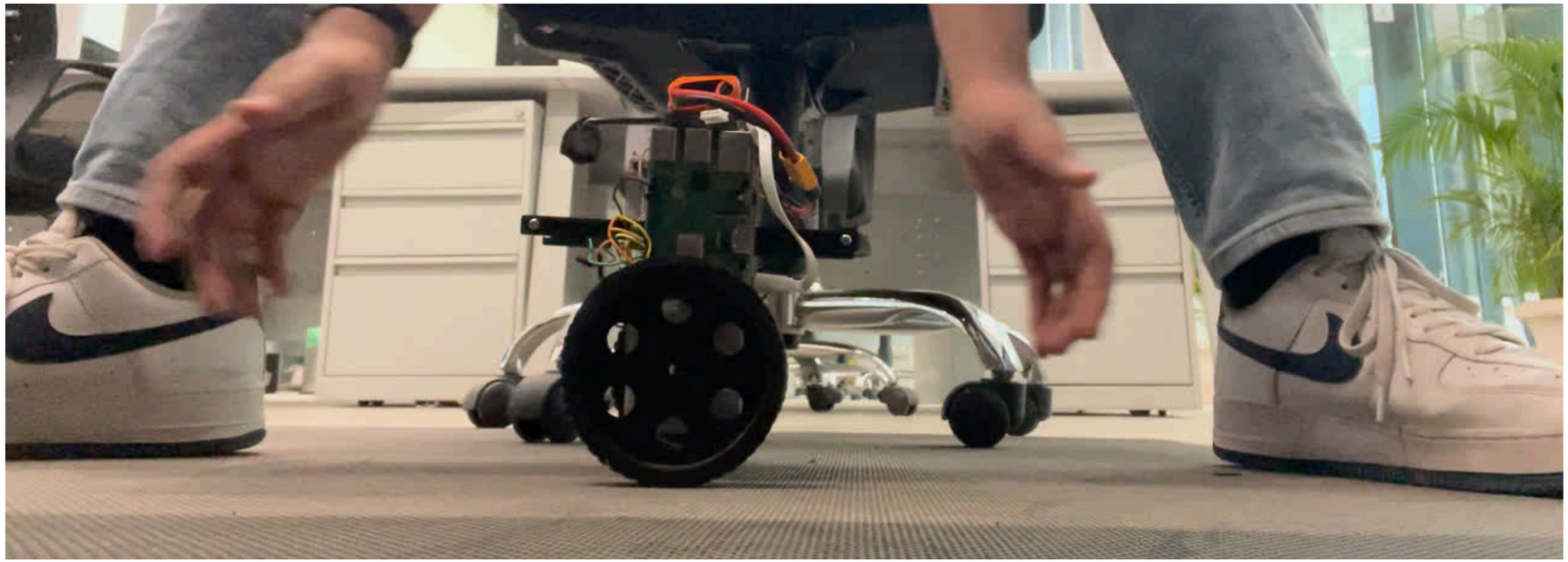
Results:

Hardware Deployment: DQN (Alpha: 0.5)



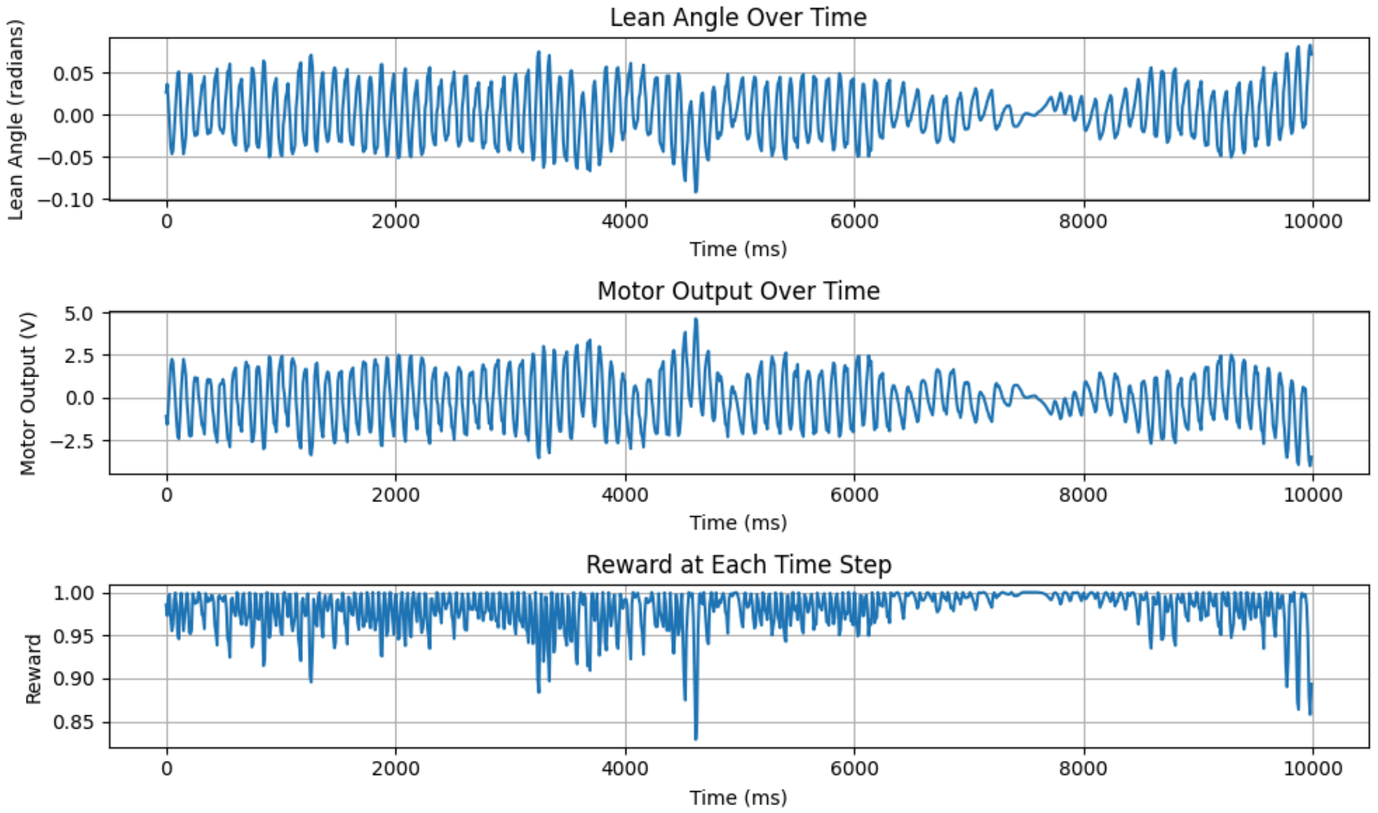
Results:

Hardware Deployment: DQN (Alpha: 0.5)



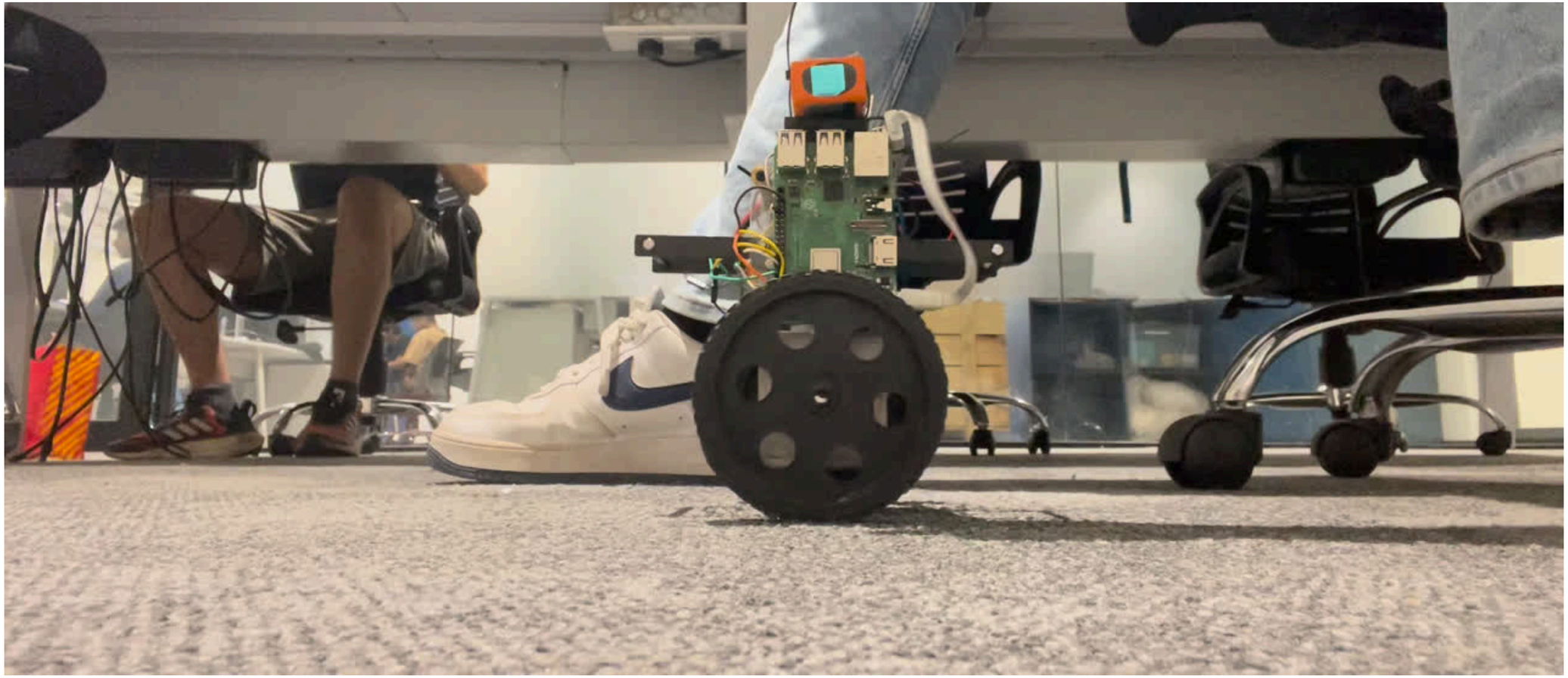
Results:

Hardware Deployment: DQN (Alpha: 0.3)



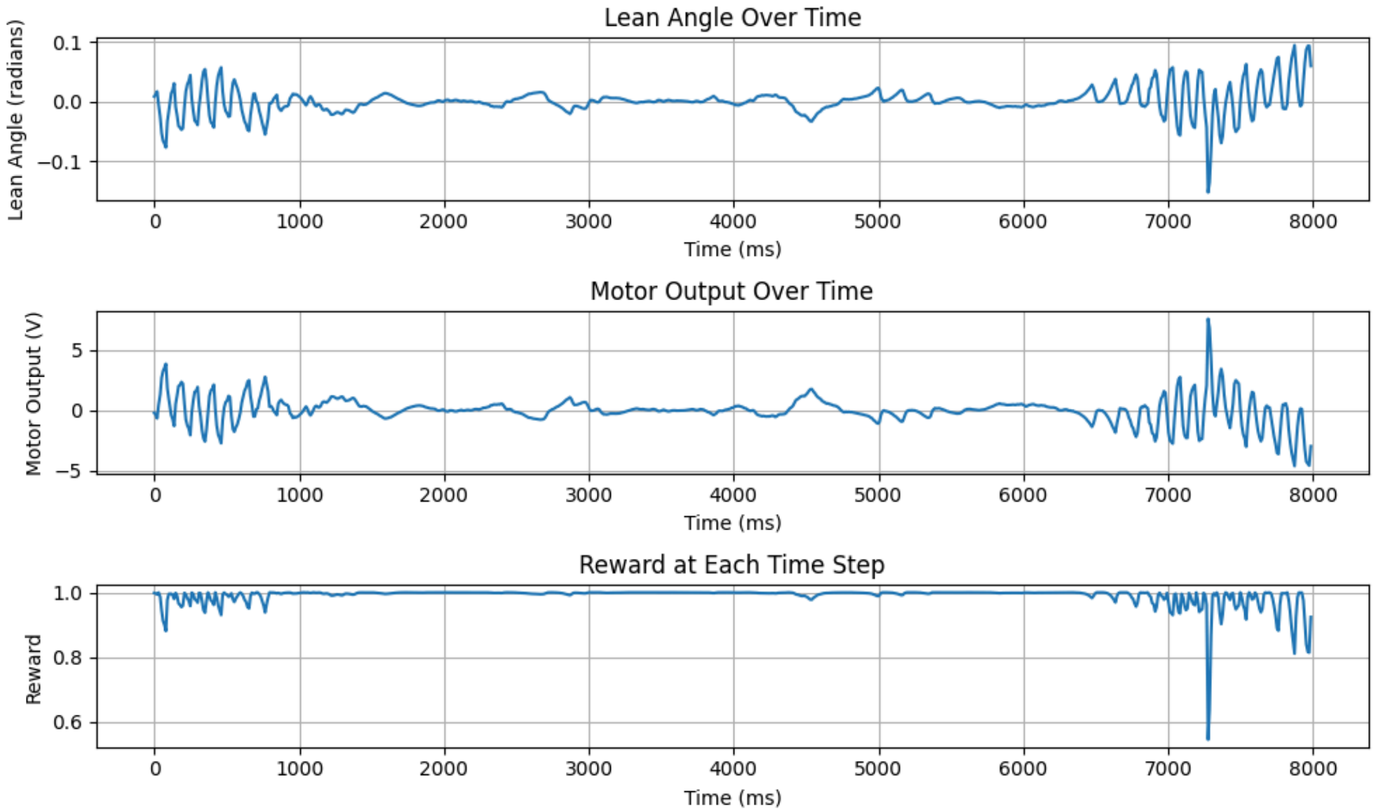
Results:

Hardware Deployment: DQN (Alpha: 0.3)



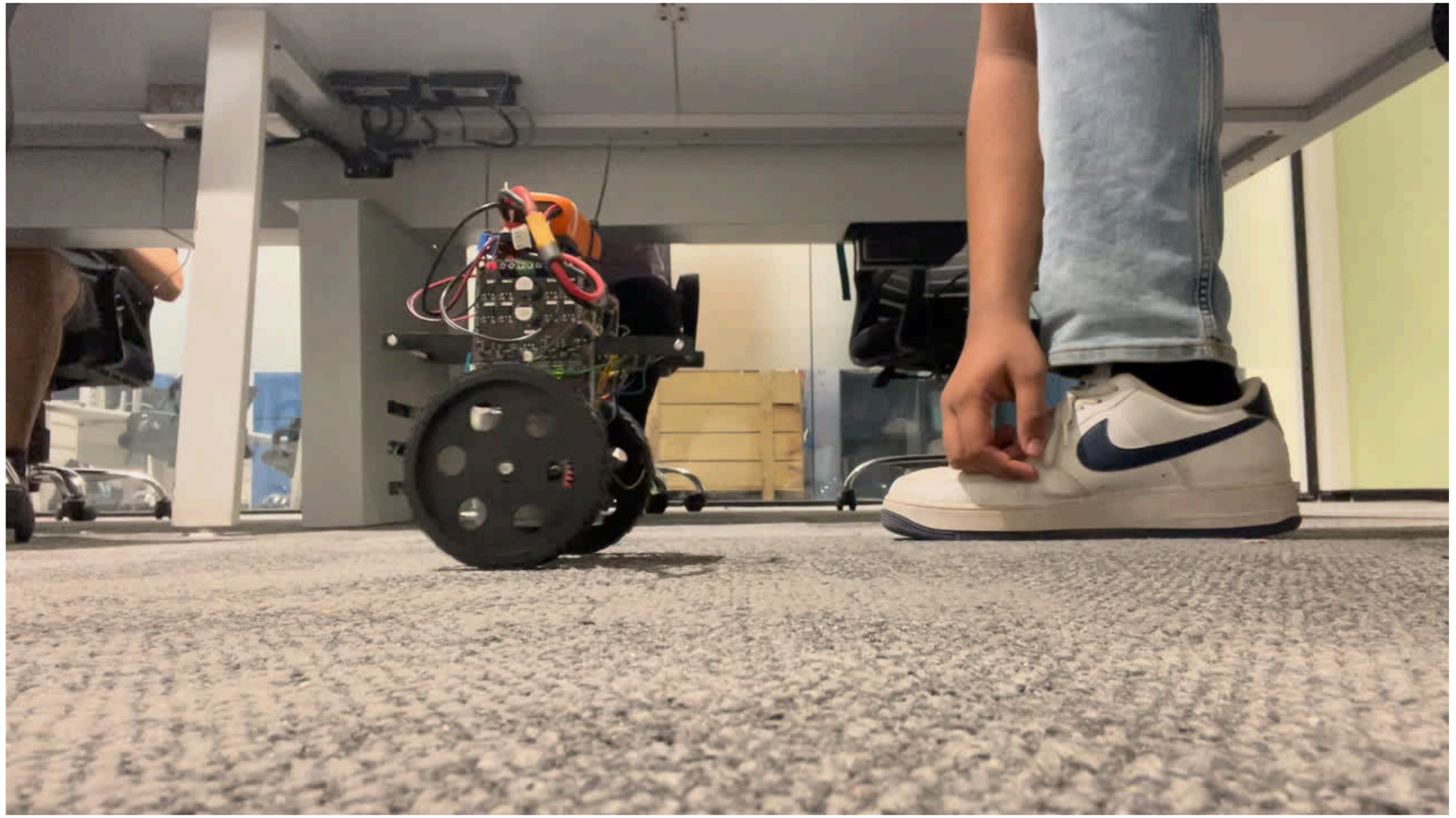
Results:

Hardware Deployment: PPO (Alpha: 0.3)



Results:

Hardware Deployment: PPO (Alpha: 0.3)



Citations

[1] Shaili Mishra and Anuja Arora. Double Deep Q Network with Huber Reward Function for Cart-Pole Balancing Problem [J]. Int J Performability Eng, 2022, 18(9): 644-653.

[2] A. d. Rio, D. Jimenez and J. Serrano. Comparative Analysis of A3C and PPO Algorithms in Reinforcement Learning: A Survey on General Environments. IEEE Access, vol. 12, pp. 146795-146806, 2024, doi: 10.1109/ACCESS.2024.3472473.

[3] E.-H. Jo and Y. Kim. Performance comparison of reinforcement learning algorithms in the CartPole game using Unity ML-Agents. Journal of Theoretical and Applied Information Technology, vol. 102, no. 16, pp. 6076–6083, Aug. 2024.