Trajectory Optimization for Active Debris Removal: a Transformer-based Reinforcement Learning Approach

Paolo Cirrincione Pazé,[1] Federico Delrio,[2] Laura Sopegno[3]

1 Department of Mechanics and Aerospace (DIMEAS) Engineering, Polytechnic of Turin, Italy
2 Department of Electronics and Telecommunication (DET), Polytechnic of Turin, Italy
3 ICT Department, University of Palermo, Italy - Electrical and Computer Engineering, University of Denver (USA)

Abstract

One of the main challenges that satellites face is the progressive accumulation of debris in LEO. Hence, the necessity to develop new strategies for debris removal, as well as for servicing and refuelling existing satellites to increase their lifespan. This poster proposes an implementation of a Deep Reinforcement Learning (DRL) framework to optimize the path of a chaser satellite, tasked with retrieving space debris or servicing other spacecrafts. Experiments have been conducted in a simulated environment, in the presence of one space debris. The proposed approach addresses imperfect environmental modelling and measurements by using a Partially Observable Markov Decision Process (POMDP). It replaces hidden state information with a belief function derived from the observation history, which is processed by a Long Short-Term Memory (LSTM) to create a fixed-length sequence. This sequence is then weighted by a Transformer encoder to capture the non-linear dynamics of the signals. The resulting semantic history guides an agent employing Proximal Policy Optimization (PPO), a model-free direct policy estimation method. PPO relies on two neural networks: a critic for value estimation and an actor for policy evaluation, implemented as Multi-Layer Perceptrons (MLPs). The model considers the motion of the satellite and debris in LEO, under J2 and atmospheric drag effects. The reward function has been designed to achieve rendezvous with the debris, minimum fuel consumption and manoeuvre duration, and optimal relative velocity. The poster concludes by presenting the results obtained.

Database generation

A simulated database has been generated to train the neural networks. The chaser S/C, tasked with ADR, is assumed to be launched into an orbit with high debris density. In this poster, the S/C starts in a low equatorial orbit. The initial COEs of the chaser are then transformed, using the Python library "Poliastro", into a 7-by-1 state vector representing the instantaneous position, velocity and mass of the satellite in ECI coordinates. The initial state of the debris is then obtained by perturbing the semi-major axis, argument of perigee and true anomaly of the chaser S/C. In this way, the debris is in a coplanar orbit similar to the one of the S/C. The state vector of the debris is then propagated forward in time, with its parameters changing under the effect of the J2 and atmospheric drag perturbations, until the end of the simulation. The database is composed of the state vector of the S/C at the initial time and the state vector of the debris, at each time-step.

a [km]	e	i [deg]	Ω	ω
7160	0.02	0	0	88.5

Environment and actions

The state vector is updated at each time-step through the state transition function. The function takes as inputs the old state, the magnitude and direction of the thrust vector and the time interval, and then performs an integration of the state vector over that interval, outputting the state vector at the successive time-step. In addition to the ideal gravity and thrust acceleration, the J2 and atmospheric drag effects are considered. With regard to the propulsion system, the chaser spacecraft is equipped with a low power ion thruster that generates a low thrust with high specific impulse. The direction of the thrust vector is defined by the elevation angle θ and the azimuth angle φ. The magnitude of the thrust vector is defined by the percentage of the maximum thrust applied. These three parameters, that determine the thrust acting on the spacecraft, are extracted by the Gaussian distributions generated by the actor at each time-step.

S/C mass [kg]	T [N]	I_{sp} [s]	c [m/s]
500	0.01	4000	39240

Proximal Policy Optimization (PPO)

The actor's weights θ are updated using the clipped surrogate objective function:

$$L^{CLIP}(\theta) = E_t[min(\rho_t(\theta)\hat{A}_t, clip(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)]$$

where $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the importance sampling ratio between the new and old policies, and $\hat{A}_t$ is the advantage estimate. It is typically computed using Generalized Advantage Estimation (GAE):

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}$$

where γ is the discount factor, λ is the GAE parameter controlling the bias-variance trade-off, and δ_t is the Temporal Difference (TD) error given by:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

where r_t is the reward at time step t, $V(s_{t+1})$ is the estimated value of the next state and $V(s_t)$ is the estimated value of the current state.

$$\theta_{new} = \theta_{old} + \alpha \nabla_\theta L^{CLIP}(\theta)$$

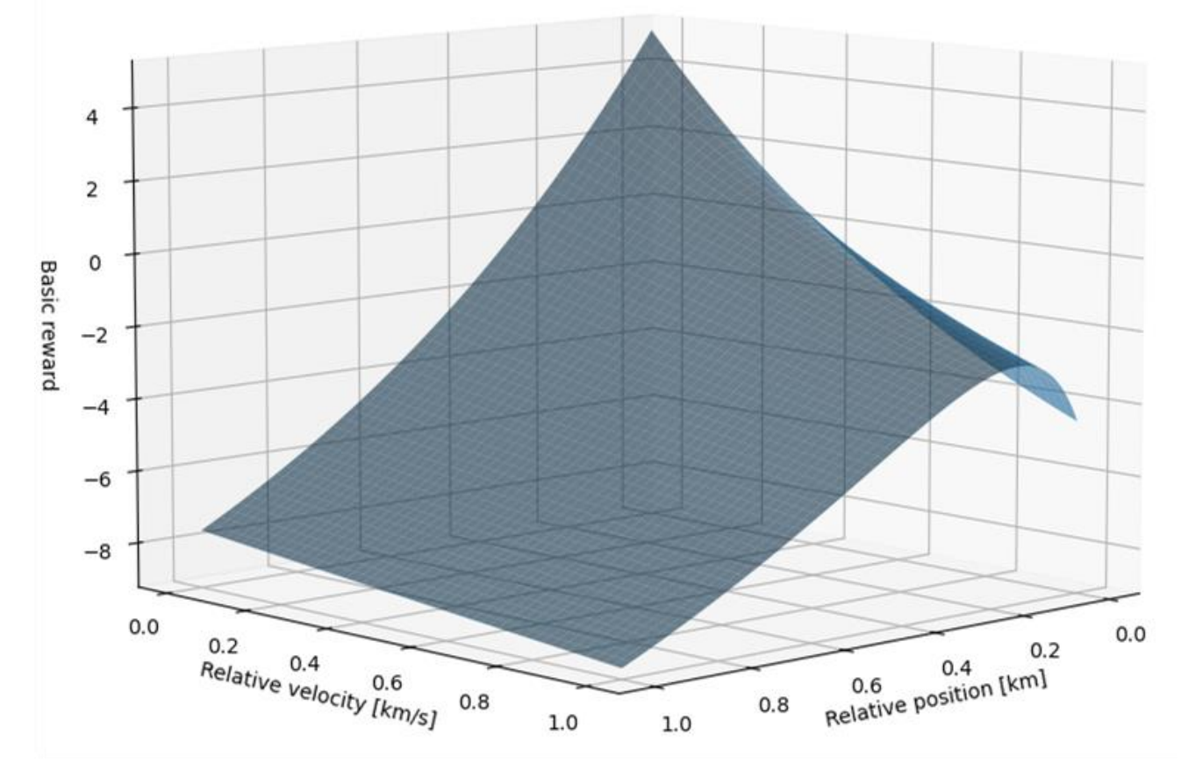
The critic's weights w are updated by minimizing the value function loss:

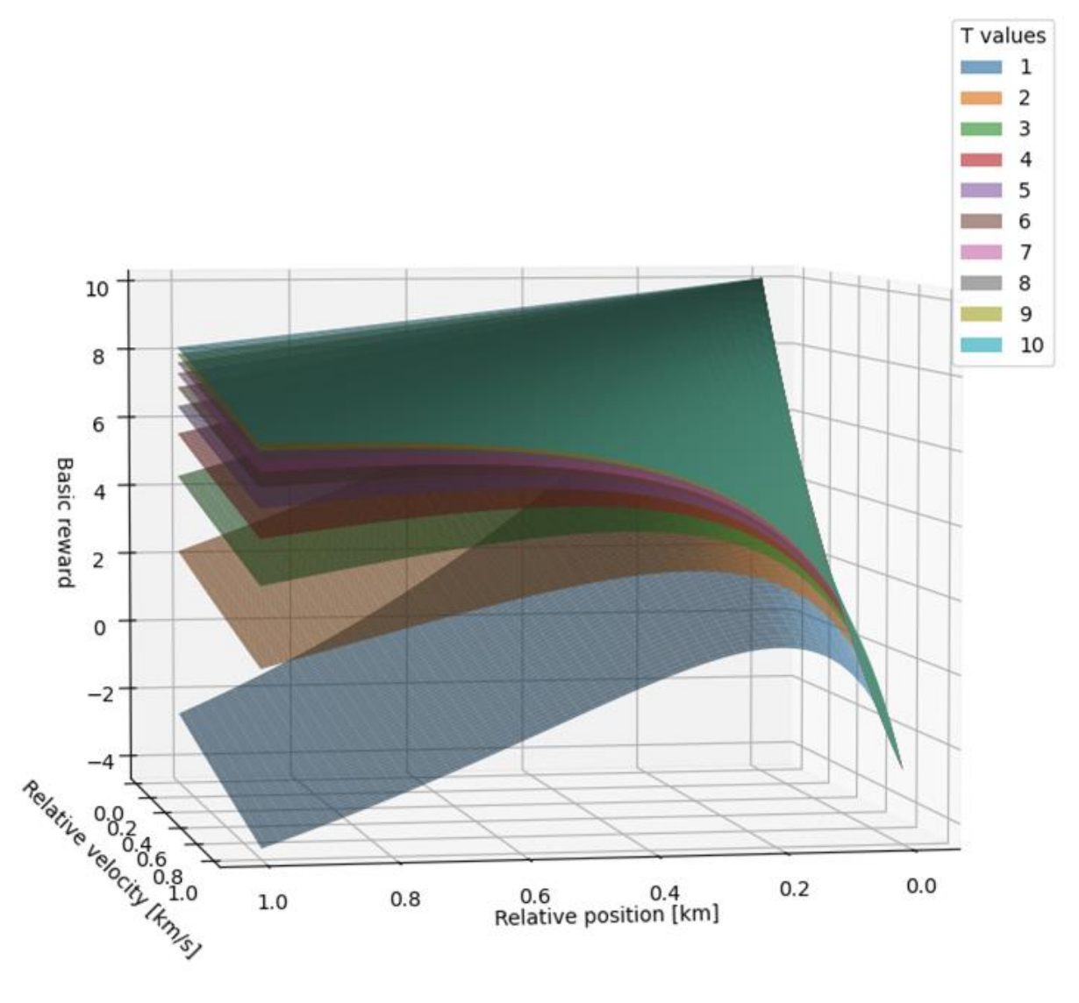
$$L^{VF}(w) = E_t\left[\left(V_w(s_t) - V_t^{target}\right)^2\right]$$

This loss represents the mean squared error between the predicted value $V_w(s_t)$ and the target value V_t^{target}, where V_t^{target} is computed using the discounted sum of rewards.

$$w_{new} = w_{old} - \beta \nabla_w L^{VF}(w)$$

Reward





$$r_{approach}(p,q) = ae^{-\frac{p}{T}} + be^{-\frac{q}{1+Kp}}$$

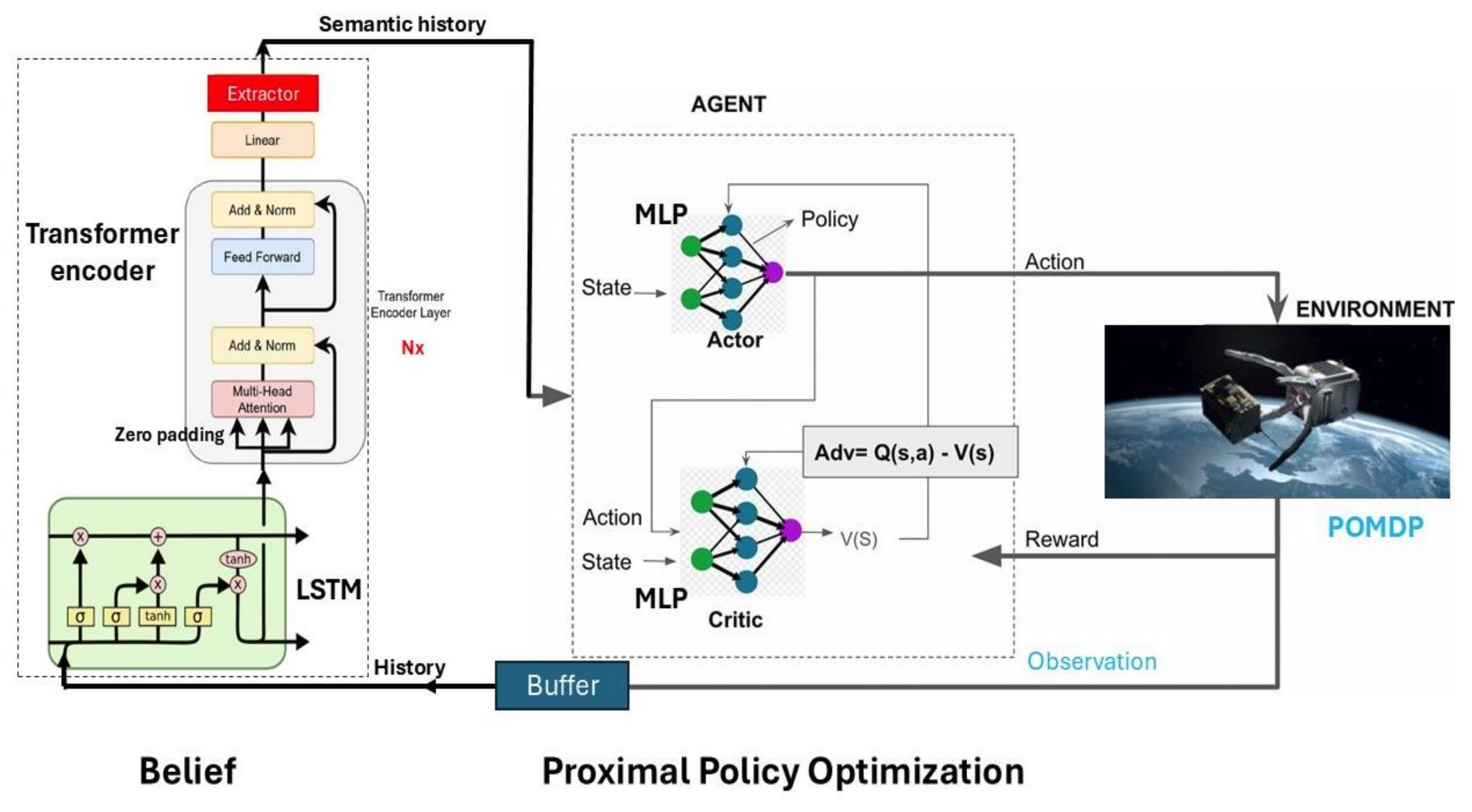
$$\nabla r_{approach} = \left(-\frac{a}{T}e^{-\frac{p}{T}} + be^{-\frac{q}{1+Kp}} \cdot \frac{qK}{(1+Kp)^2}, -\frac{b}{1+Kp}e^{-\frac{q}{1+Kp}}\right)$$

a=20, b=15, T=1, K=10

$$r(p,q,t,a_{engine}) = r_{approach} - c\ t - d\ a_{engine}$$

POMDP

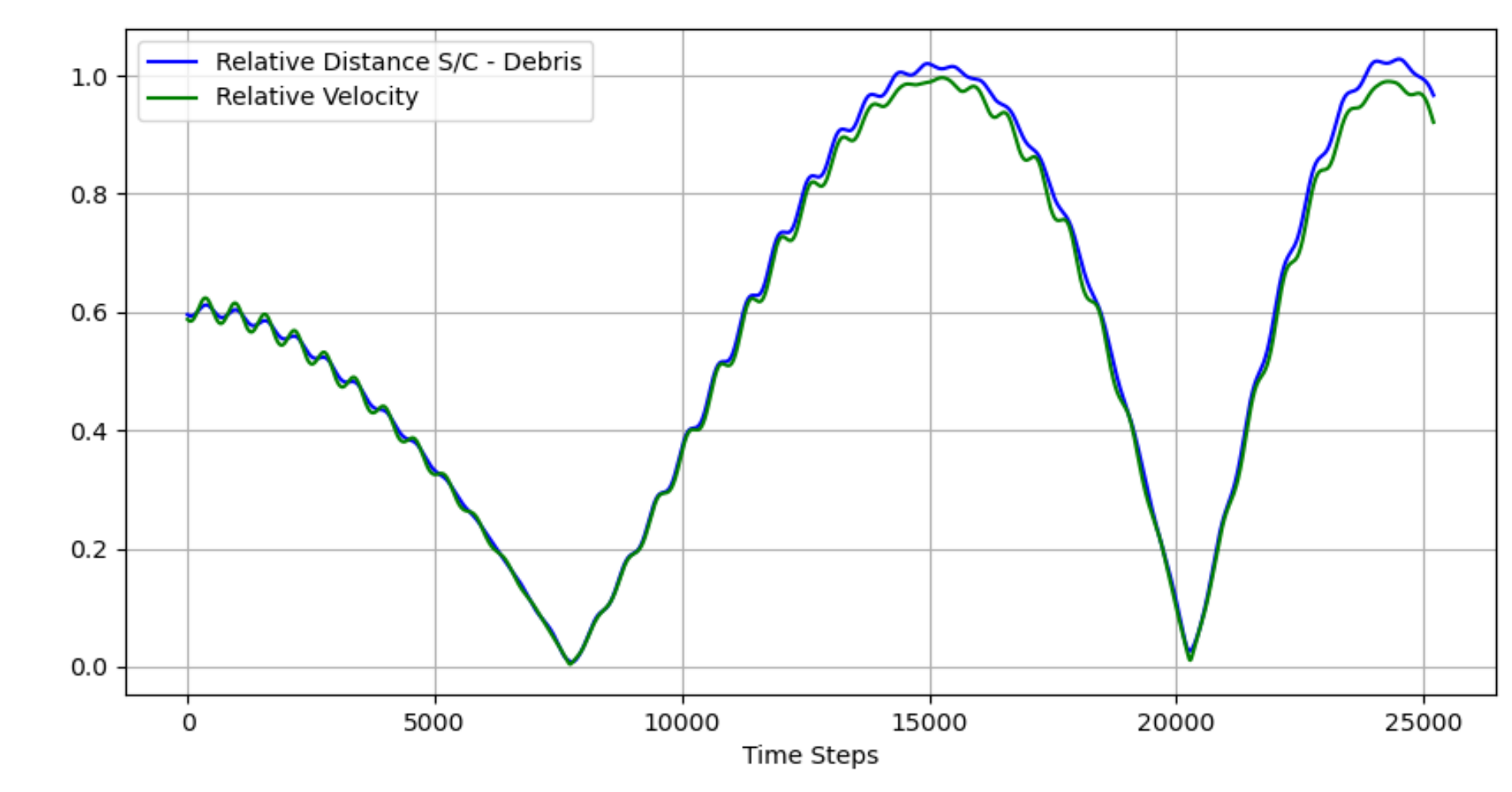
In this POMDP-based architecture, the agent makes decisions based on partial observations of the environment, such as in space debris removal where sensing limitations prevent precise knowledge of debris' position or velocity. To address this, the architecture maintains a belief over the environment's state, updated using a sequence of observation vectors processed by an LSTM and a transformer encoder. The LSTM handles short-term dependencies and addresses the vanishing gradient issue for moderately sized sequences, producing a fixed-length weighted sequence. The transformer encoder refines this sequence using long-term correlations through multi-head self-attention. During training, zero-padding ensures uniform input length, while in testing, the transformer's final output is passed to Proximal Policy Optimization (PPO). The agent's PPO implementation uses Multilayer Perceptrons (MLPs) for both the actor and critic, with two hidden layers. The actor outputs the angles and engine thrust lever (continuous actions), modeled using Gaussian distributions.



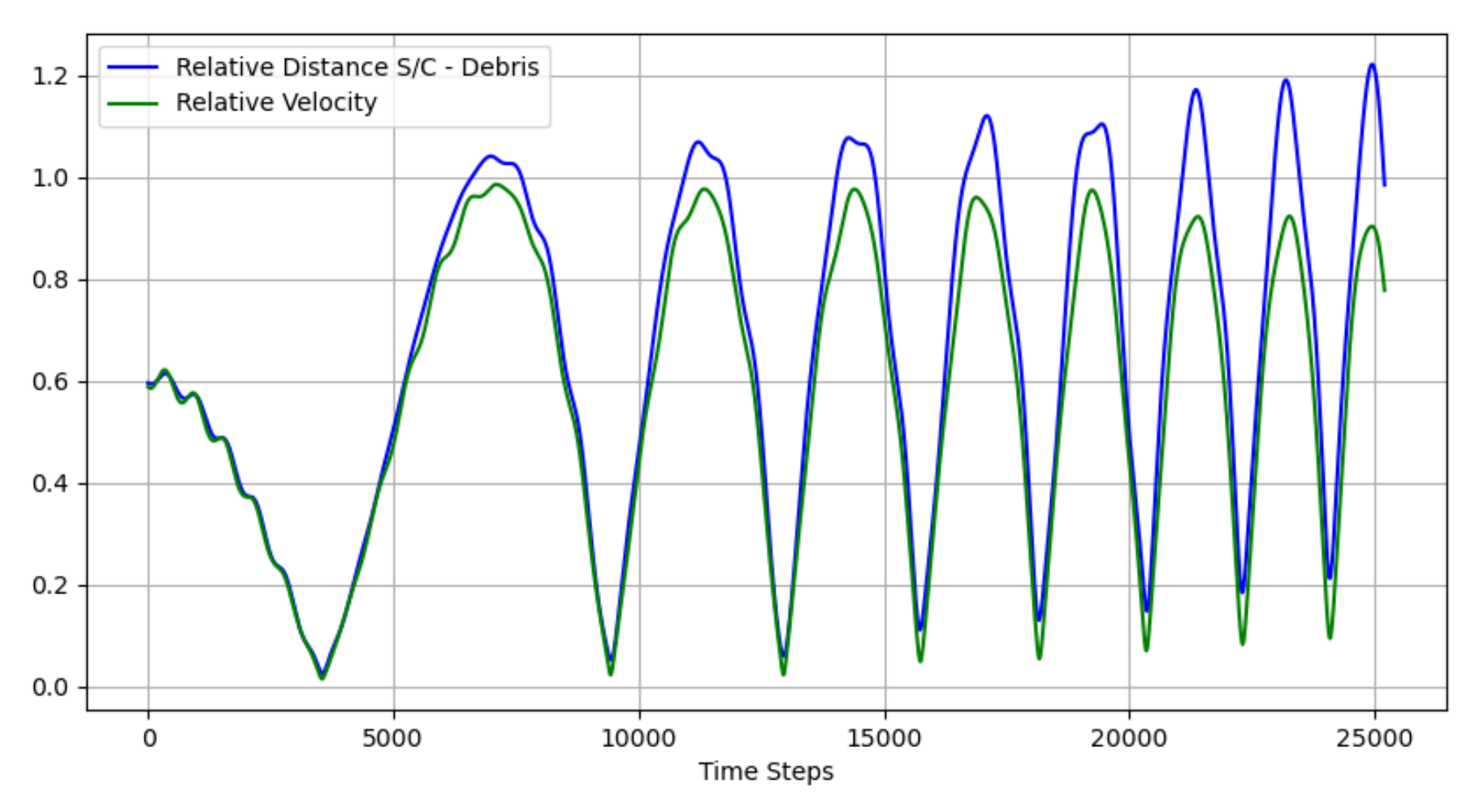
This research is part of the BrS-AI-ECG project, funded by the Italian Ministry of Foreign Affairs and International Cooperation. Dr. Delrio acknowledges funding from Eurecom.

Results

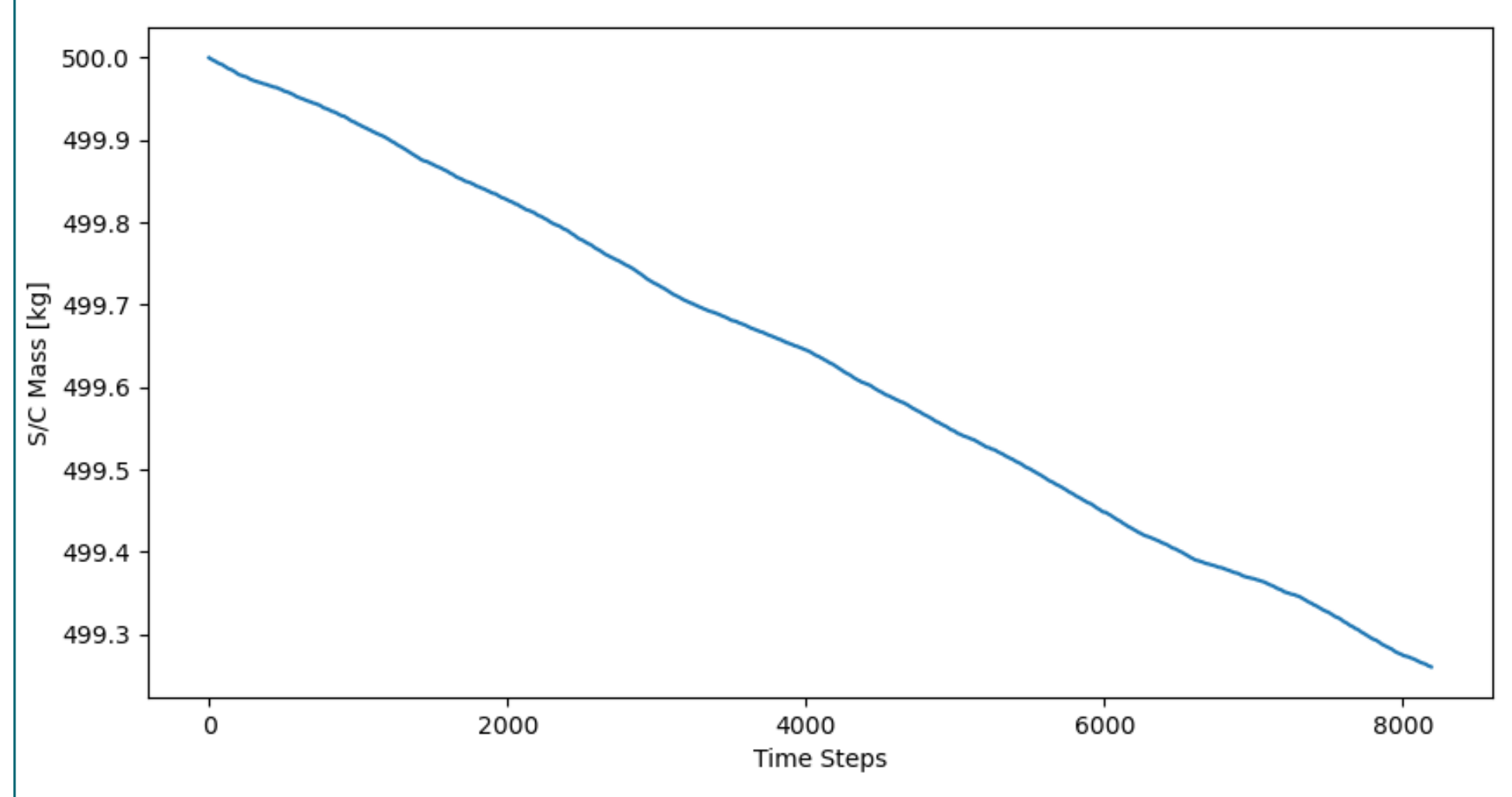
Training (stochastic actions choice)



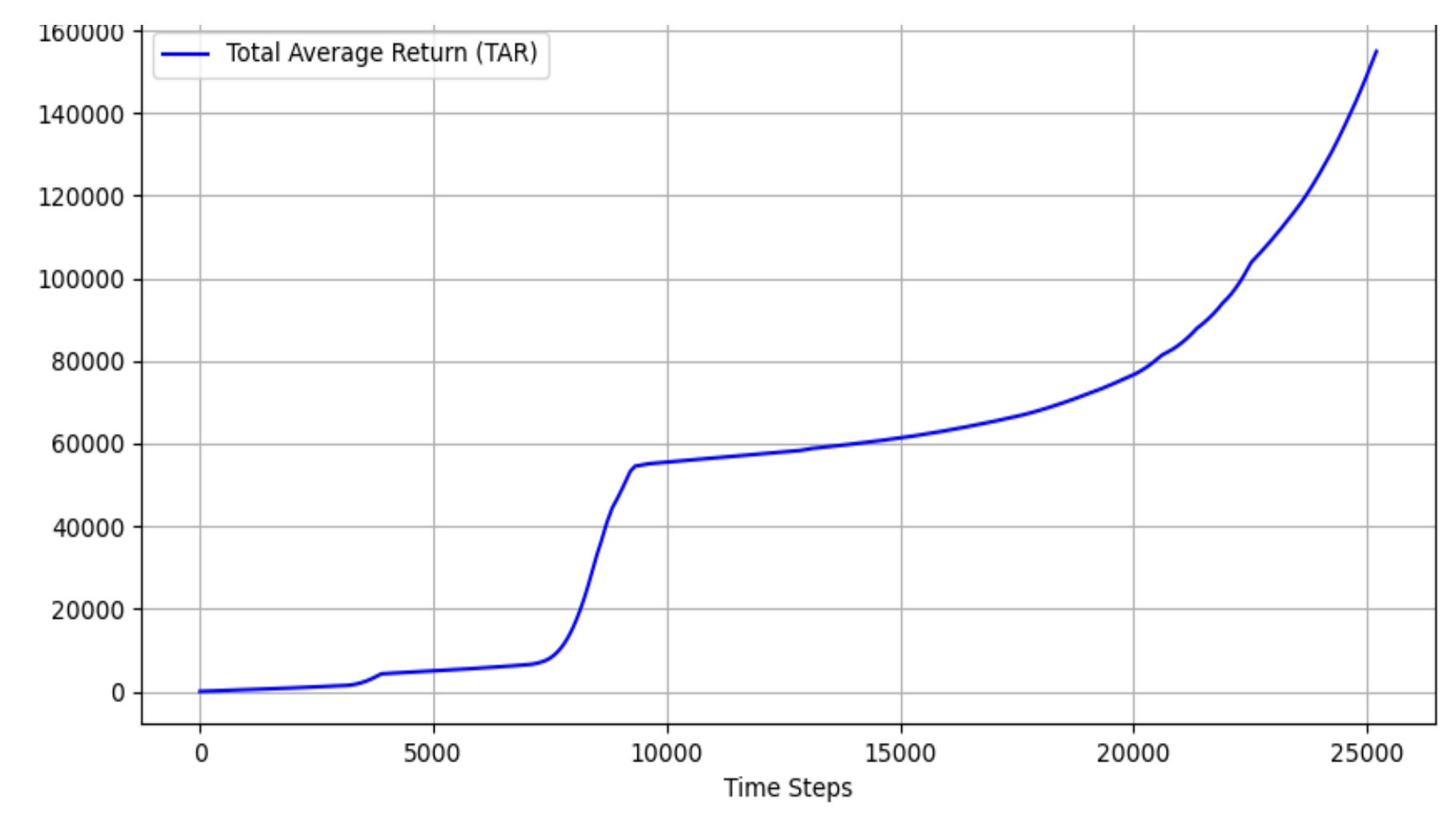
Test (deterministic actions choice)



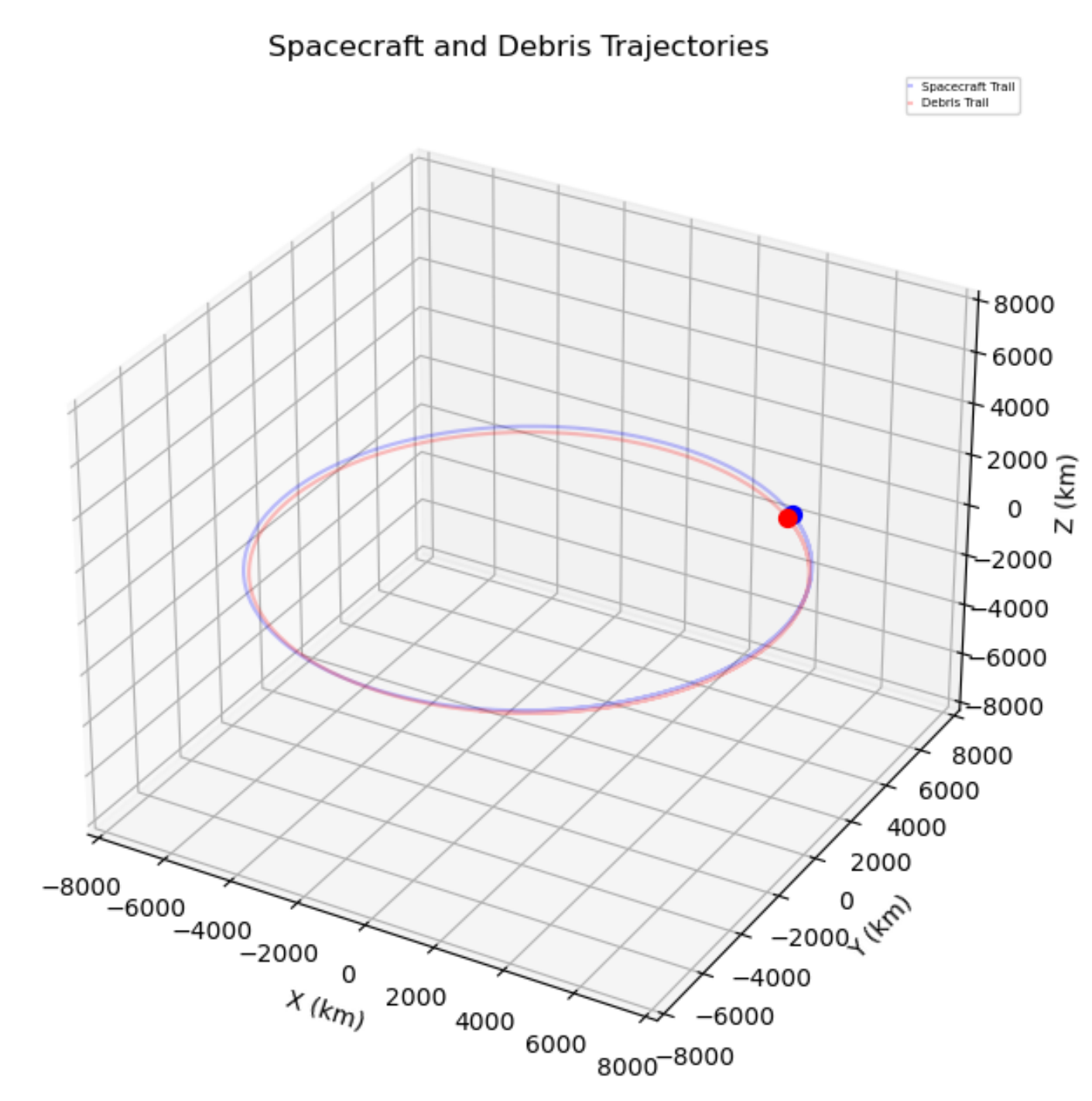
Spacecraft mass

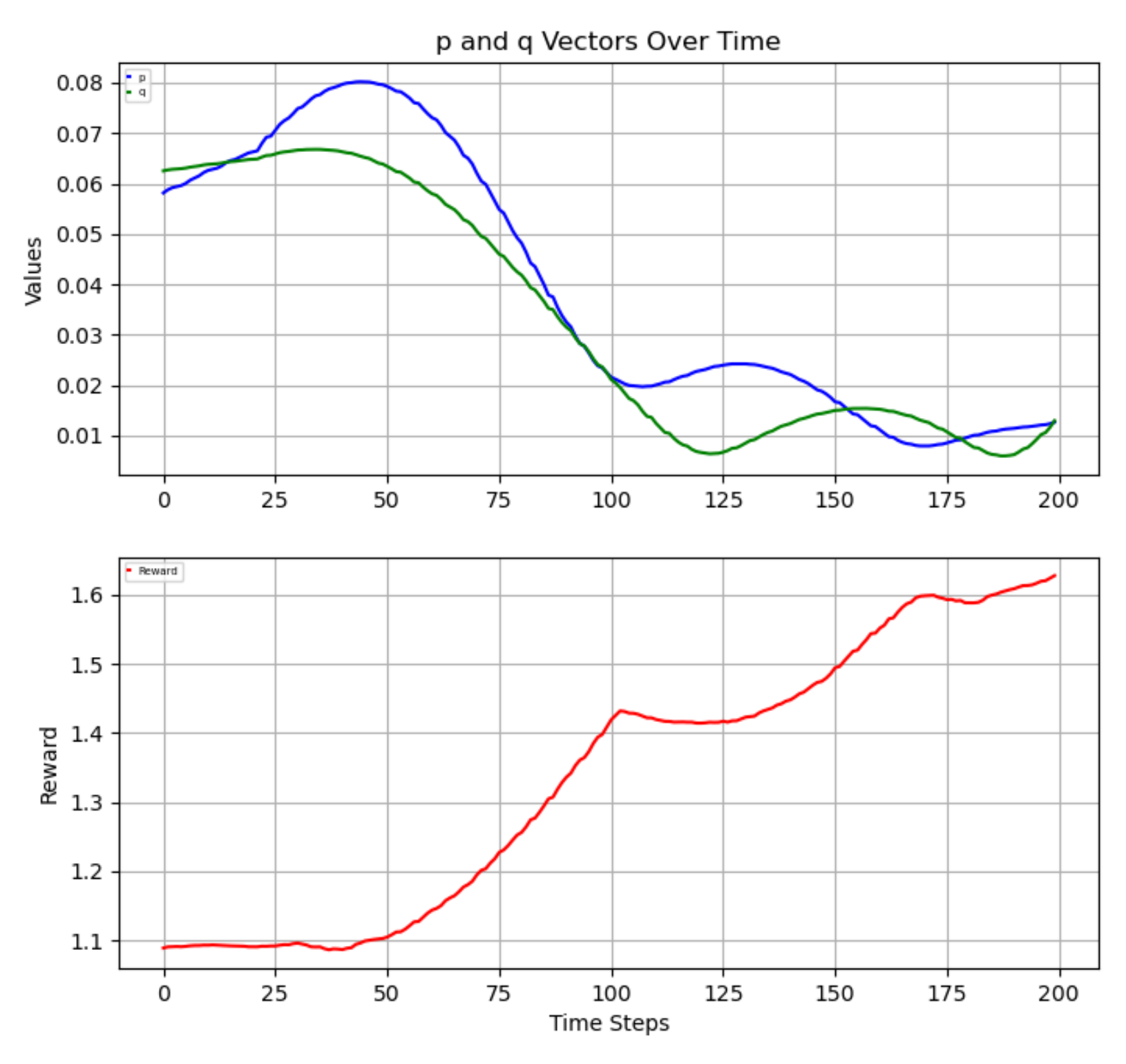


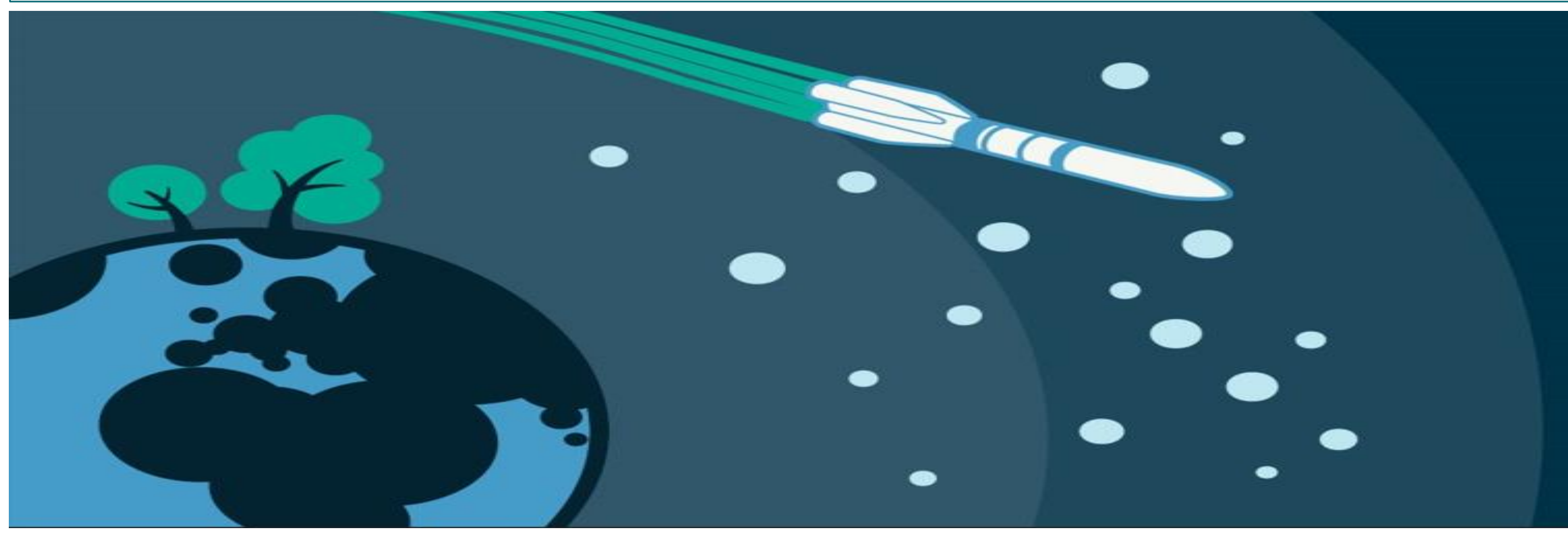
Total Average Return



3D and 2D trajectories







CLEAN SPACE DAYS 2024

8th to 11th October 2024

Poster Session

#CSD2024