Enhancing Spacecraft Trajectory Optimization Using LSTM-Augmented Reinforcement Learning

Evan Coleman
University of Mary Washington
Fredericsburg, VA
ecolema4@umw.edu

ABSTRACT

Spacecraft trajectory optimization presents a challenging control problem due to the complex, nonlinear dynamics governing orbital mechanics and the high-dimensional continuous action space associated with thruster control. This paper explores the ability of reinforcement learning agents to handle partially observable environments. We demonstrate that the temporal modeling capabilities of LSTMs improve the agent's ability to plan satellite maneuvers. Using the high-fidelity Basilisk astrodynamics library, we demonstrate that a PPO agent augmented with an LSTM not only achieves significantly higher performance in a partially observable satellite attitude control task but also quantitatively establishes the advantage of recurrent memory over a memory-less agent in this specific domain..

ABOUT THE AUTHORS

Evan Coleman is an Assistant Professor of Computer Science at the University of Mary Washington in Fredericksburg, Virginia. He holds an MS in Mathematics from Syracuse University and a PhD in Modeling and Simulation from Old Dominion University. Prior to joining the faculty at the University Mary Washington, he spent 13 years in industry, primarily working with the Department of Defense. His research interests include high-performance computing and large-scale simulations, as well as the application of artificial intelligence and machine learning to those fields.

Enhancing Spacecraft Trajectory Optimization Using LSTM-Augmented Reinforcement Learning

Evan Coleman
University of Mary Washington
Fredericksburg, VA
ecolema4@umw.edu

INTRODUCTION

Reinforcement Learning (RL) has emerged as a powerful approach for aerospace missions because it excels at decision-making under uncertainty and over long time horizons (Forestieri and Casalino, 2025). Unlike traditional controllers that react myopically to current errors, RL agents learn policies that consider long-term rewards, enabling a form of planning or "situational awareness" beyond the immediate step (Elkins et al., 2020). This is especially useful in space operations where sequences of actions (e.g. a series of thruster burns or attitude slews) must be optimized together to achieve an end goal. Another key advantage is that RL does not require an exact physics model of the spacecraft or environment; it can learn control policies model-free by trial and error in simulation (Loettgen et al., 2023). Studies report that deep RL controllers can even outperform standard algorithms; for instance, a PPO-based attitude controller was shown to outperform a classical quaternion feedback controller in simulation tests (Elkins et al., 2020).

Attitude control is a fundamental task for satellites – ensuring the spacecraft points in the right direction for communication, imaging, or power. Traditionally, this is handled by feedback controllers like PD/PID loops, quaternion feedback (e.g. QRF controllers), or LQR regulators. Such controllers are reliable and computationally cheap, but they must be carefully tuned and often assume linearized or fixed dynamics. This application area has been studied before (Elkins et al., 2020; Shi et al., 2022; Stephenson et al., 2024), and the goal of this work is to expand upon those efforts by investigating the ability of recurrent neural networks (RNNs) to improve the ability of the agent in a partially observed environment.

The main technique explored here is Long Short-Term Memory (LSTM). LSTM-enhanced reinforcement learning algorithms demonstrate significant advantages over standard methods in partially observable environments, particularly in aerospace, space, and defense domains where temporal memory and sequential decision-making are critical (Meng et al., 2021). LSTM networks address fundamental limitations in standard RL by extending beyond the Markovian assumption through sophisticated memory mechanisms. Standard reinforcement learning struggles when environmental states are hidden, sensors provide incomplete information, or critical data is distributed across multiple time steps. LSTM networks excel by maintaining compressed observation histories through hidden states that encode relevant temporal dependencies while filtering noise through selective gating mechanisms (Duell et al., 2012; Meng et al., 2021).

Adding LSTM memory to RL agents is expected to yield several benefits. First, the policy can learn temporal strategies: for example, it may recall how long it has been thrusting and adjust future thrust accordingly, rather than making decisions solely on instantaneous state. This can smooth the control profile and reduce oscillations. Second, the LSTM's internal state can act as a surrogate for unmodeled dynamic variables or disturbances (e.g. slight modeling errors in gravity), helping the agent adapt. Empirical results in analogous domains (e.g. automotive control (Yang et al., 2024)) show that recurrent agents achieve higher rewards and fewer collision events than feedforward ones.

However, these advantages come with limitations. The LSTM variants of RL algorithms tend to have many more parameters, making training slower and more data-intensive. Recurrent networks can be also harder to tune: issues like vanishing/exploding gradients may arise despite LSTM's gating structure proceedings (Bakker, 2001). Additionally, if the episode lengths vary widely, zero-padding or truncated sequences must be handled carefully. Finally, the agent is still model-free, so it may struggle with highly precise maneuvers where model-based optimal control excels.

In this study, the focus is on exploring the ability of memory (in the form of an LSTM) to improve the performance of the Proximal Policy Optimization (PPO) algorithm. In order to investigate the effect of aiding these RL approaches with an LSTM, a satellite attitude control mission is created using the Basilisk Reinforcement Learning framework (BSK-RL) (Stephenson and Schaub, 2024) which utilizes the Basilisk spacecraft simulation framework (Kenneally et al., 2020). Basilisk is a modular, high-fidelity simulation environment that offers easy integration to reinforcement learning environments such as Gymnasium (Towers et al., 2024) and Stable-Baselines3 (Raffin et al., 2021). This style of mission has been used in previous RL efforts (Stephenson et al., 2024; Shi et al., 2022) due to the inherent partial observability, and the direct analog to more complex missions. The results of this investigation show that the use of an LSTM in conjunction with the PPO method can improve the performance of the reinforcement learning agent being trained. Future efforts are planned to continue to explore the affect that partial observability can have on the application of reinforcement learning algorithms to satellite tasking missions.

This paper is structured as follows: first, related work is presented, followed by a brief background on the relevant concepts of Markov Decision Processes, Proximal Policy Optimization, and Long Short-Term Memory networks. Then the simulated spacecraft environment and the specific configurations for our fully and partially observable experiments are described. Subsequently, results are presented along with a comparative analysis, followed by a discussion of the findings. Finally, the paper concludes with a summary of contributions and suggested directions for future research.

RELATED WORK

The survey by Hermann and Schaub (Herrmann and Schaub, 2023) provides a great overview of the use of RL for spacecraft trajectory optimization. There are many investigations into the use of RL in this domain, and recent studies have begun exploring the incorporation of time-sensitive data into an RNN. Recent research has successfully applied deep RL to agile satellite attitude maneuvers. For example, agents trained with Proximal Policy Optimization or Deep Deterministic Policy Gradient (DDPG) have achieved high-accuracy pointing during large-angle slews using only discrete thruster or reaction wheel actions Elkins et al. (2020).

Of the many approaches towards using reinforcement learning to aid in spacecraft mission optimization, Gaudet et al (Gaudet et al., 2020a,b,c) use meta-RL – both with and without RNNs – to attempt to find develop guidance systems for spacecraft across different scenarios, Kolosa (Kolosa, 2019) uses a DDPG with Prioritized Experience Replay (PER) as well as a novel parameter initialization technique, the work by Scorsoglio et al (Scorsoglio et al., 2022) uses image based data along with an RNN, and the study by Yan et al leverages a hierarchical PPO architecture to enable missiles to simultaneously guide toward targets and evade interceptors using two-layer structures with LSTM-enhanced policy selectors (Yan et al., 2022).

The work by Hovell and Ulrich (Hovell and Ulrich, 2021) models spacecraft operations using a variant of the Deep Deterministic Policy Gradient method, but without any sort of recurrent neural network to better capture temporal information. More directly related, the work by Shi et al (Shi et al., 2022) models a satellite attitude control systems using TD3-LSTM (Twin Delayed Deep Deterministic Policy Gradient with LSTM) demonstrate real-world success in moving target tracking scenarios. The LSTM component learns target movement patterns from image positions, enabling real-time desired attitude generation while optimizing energy consumption—a critical constraint in space operations. Other similar efforts include the work by Wang et al (Wang et al., 2024) who investigate techniques for intercept guidance against maneuvering targets under partial observability.

BACKGROUND

Markov Decision Processes

A Markov Decision Process (MDP) provides a mathematical framework for modeling sequential decision-making problems under uncertainty. Formally, an MDP is defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where:

- $\mathcal{S}$ is a finite set of states representing all possible configurations of the system
- $\mathcal{A}$ is a finite set of actions available to the decision-making agent
- $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition probability function, where $\mathcal{P}(s'|s, a)$ represents the probability

of transitioning to state s' given current state s and action a
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, providing immediate feedback for taking action a in state s
- $\gamma \in [0, 1]$ is the discount factor, determining the relative importance of immediate versus future rewards

The fundamental assumption underlying MDPs is the Markov property, which states that the future evolution of the system depends only on the current state and action, independent of the history of states and actions that led to the current state. Mathematically, this is expressed as:

$$P(S_{t+1} = s'|S_t = s, A_t = a, S_{t-1}, A_{t-1}, \ldots, S_0, A_0) = P(S_{t+1} = s'|S_t = s, A_t = a) \tag{1}$$

The objective in an MDP is to find an optimal policy $\pi^* : \mathcal{S} \to \mathcal{A}$ that maximizes the expected cumulative discounted reward $V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \mid S_0 = s\right]$.

In the context of satellite simulation, MDPs provide a natural framework for modeling autonomous satellite operations, constellation management, and mission planning scenarios. The adaptation involves careful design of the state space, action space, and reward structure to capture the essential dynamics of satellite systems. For a single satellite mission, the state space $\mathcal{S}$ typically encompasses the satellite's orbital parameters, onboard resource levels, and operational modes. Specifically, a state $s \in \mathcal{S}$ might be represented as:

$$s = (p, v, \theta, E, M, D, T) \tag{2}$$

where p denotes the satellite's position vector, v represents velocity, θ captures attitude parameters, E indicates energy levels, M represents memory utilization, D denotes data storage, and T reflects thermal conditions. The action space $\mathcal{A}$ for satellite operations commonly includes attitude control maneuvers, thruster firings, communication scheduling, data collection commands, and power management decisions.

The transition probabilities $\mathcal{P}(s'|s, a)$ incorporate orbital mechanics, environmental perturbations, and system dynamics. For orbital propagation, these transitions often integrate gravitational forces, atmospheric drag, solar radiation pressure, and third-body perturbations. The stochastic nature accounts for uncertainties in environmental models, measurement noise, and actuator performance. The reward function $\mathcal{R}(s, a)$ is designed to reflect mission objectives and operational constraints. For Earth observation missions, rewards might emphasize data quality and coverage. However, selecting an appropriate reward for reinforcement learning is often difficult and needs to be tailored to each individual application.

Partially Observable Markov Decision Processes

While MDPs assume complete state observability, many practical scenarios involve incomplete or noisy state information. Partially Observable Markov Decision Processes (POMDPs) extend the MDP framework to handle such situations. A POMDP is formally defined as a tuple $\mathcal{M}_{\text{POMDP}} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \Omega, \mathcal{O}, \gamma)$, where the first five elements are identical to the MDP formulation, with additional components:

- Ω is a finite set of observations that the agent can receive
- $\mathcal{O} : \mathcal{S} \times \mathcal{A} \times \Omega \to [0, 1]$ is the observation probability function, where $\mathcal{O}(o|s', a)$ represents the probability of observing o after taking action a and transitioning to state s'

Since the true state is not directly observable, the agent must maintain a belief state b_t, which is a probability distribution over possible states:

$$b_t(s) = P(S_t = s|o_1, a_1, o_2, a_2, \ldots, o_t) \tag{3}$$

and the belief state is updated using Bayes' rule after each observation.

POMDPs are particularly well-suited for satellite simulation scenarios where state estimation uncertainty is significant. Common sources of partial observability in satellite systems include sensor limitations, communication delays, environmental noise, and intermittent ground contact windows. In satellite applications, the observation space Ω typically consists of sensor measurements, telemetry data, and ground-based tracking information. For a remote sensing

satellite, observations might include:

$$o = (r_{\text{GPS}}, a_{\text{IMU}}, T_{\text{thermal}}, P_{\text{power}}, S_{\text{star}}) \tag{4}$$

where r_{GPS} represents GPS position measurements (when available), a_{IMU} denotes inertial measurement unit readings, T_{thermal} captures thermal sensor data, P_{power} reflects power system telemetry, and S_{star} represents star tracker observations.

The observation probability function $\mathcal{O}(o|s', a)$ models the relationship between true system states and available measurements. Belief state maintenance becomes crucial for satellite operations under partial observability. The belief state must integrate orbital dynamics models with measurement updates while accounting for model uncertainties and observation noise. The POMDP framework enables sophisticated autonomous satellite behaviors, including adaptive sensor scheduling based on belief state uncertainty, robust trajectory planning under state estimation errors, and fault detection and isolation strategies.

Proximal Policy Optimization

Proximal Policy Optimization (PPO) (Schulman et al., 2017) addresses the challenge of stable policy updates in reinforcement learning by constraining policy changes to a trusted region. It is a successor to the Trusted Region Policy Optimization (TRPO) (Schulman et al., 2015). Both policies can be used for either discrete or continuous actions spaces. PPO in particular has been applied to many different areas, including spacecraft optimization problems (Elkins et al., 2020; Yan et al., 2022; Stephenson et al., 2024).

PPO optimizes a clipped surrogate objective function, L, that prevents excessively large policy updates:

$$L(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) A(s,a), \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) A(s,a) \right) \right] \tag{5}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio between new and old policies, $A(s,a)$ represents the advantage estimate, and ϵ is the clipping parameter controlling the policy update magnitude. The advantage function can be implemented different ways, but is meant to capture whether a given action from some state, (s,a), is better or worse than the other actions the agent could have taken from that state.

PPO alternates between collecting trajectory data using the current policy and performing multiple epochs of minibatch optimization on the collected data. PPO's stability and sample efficiency make it well-suited for complex satellite mission planning scenarios where exploration must be carefully managed to avoid unsafe configurations. The algorithm excels in discrete action spaces common in high-level mission planning tasks. PPO's clipped objective function prevents policy degradation during training, which is crucial for satellite applications where policy rollbacks may not be feasible. The entropy regularization encourages exploration of diverse mission strategies while maintaining operational safety through conservative policy updates.

Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) networks address the vanishing gradient problem in traditional recurrent neural networks, making them exceptionally well-suited for processing the long-duration time series data characteristic of satellite operations. LSTM networks employ a sophisticated gating mechanism to selectively retain and forget information over extended temporal sequences. LSTM networks excel in applications requiring temporal pattern recognition, anomaly detection, and predictive modeling. The extended memory capabilities enable processing of orbital dynamics, thermal cycles, and long-term degradation patterns spanning multiple orbital periods.

The combination of LSTM networks with reinforcement learning algorithms creates powerful frameworks for satellite control under partial observability and temporal dependencies. LSTM-enhanced policy networks can maintain memory of past observations and actions, enabling more informed decision-making in scenarios with delayed consequences or intermittent observations.

ENVIRONMENT SIMULATION

In this study, the Basilisk astrodynamics library (Kenneally et al., 2020) is used to simulate spacecraft dynamics. This environment simulates a spacecraft attitude control problem where the satellite is attempting to point towards a given target direction. Because this scenario has the property of partial observability, it can demonstrate the advantages of recurrent policies over memory-less ones in sequential decision-making tasks.

The complete spacecraft state $\mathbf{x} \in \mathcal{X}$ consists of $\mathbf{x} = [\mathbf{q}, \boldsymbol{\omega}, \mathbf{t}]^T$, where:

- $\mathbf{q} = [q_1, q_2, q_3, q_0]^T \in \mathbb{R}^4$ is the attitude quaternion (unit norm)
- $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T \in \mathbb{R}^3$ is the angular velocity in body frame (rad/s)
- $\mathbf{t} = [t_x, t_y, t_z]^T \in \mathbb{R}^3$ is the target direction unit vector in inertial frame

The agent observes only a subset of the state, creating partial observability: $\mathbf{o} = [\boldsymbol{\omega}, \mathbf{t}, \theta_e]T \in \mathcal{O} \subset \mathbb{R}^7$, where:

- $\boldsymbol{\omega} \in [-5, 5]^3$ rad/s is the angular velocity
- $\mathbf{t} \in [-1, 1]^3$ is the target direction (unit vector)
- $\theta_e \in [0, \pi]$ is the pointing error magnitude

Crucially, the current attitude $\mathbf{q}$ is *not* observable, requiring the agent to integrate angular velocity over time to estimate orientation.

The control input is a 3D torque vector:

$$\mathbf{u} = \boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]^T \in [-\tau_{max}, \tau_{max}]^3 \tag{6}$$

where $\tau_{max} = 1.0$ N·m.

The spacecraft dynamics follow Euler's equations for rigid body rotation.

Reward Function

The reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is:

$$r(s, a) = r_{point} + r_{progress} + r_{vel} + r_{control} + r_{success} + r_{osc} \tag{7}$$

where:

- $r_{point} = \max(0, 1 - \theta_e/\pi)$ - pointing accuracy reward
- $r_{progress} = 0.2 \cdot \max(0, \Delta\theta_e)$ - improvement bonus
- $r_{vel} = -0.05\|\boldsymbol{\omega}\|^2$ - angular velocity penalty
- $r_{control} = -0.01\|\mathbf{u}\|^2$ - control effort penalty
- $r_{success} = 5.0 \cdot \mathbb{1}[\theta_e < \theta_{tol}]$ - success bonus
- $r_{osc} = -1.0 \cdot \mathbb{1}[\theta_e < 0.3 \wedge \|\boldsymbol{\omega}\| > 0.5]$ - oscillation penalty

The reward components were selected based on prior work in attitude control and tuned through empirical observation. A formal sensitivity analysis is planned to explore how scaling or omitting terms affects convergence and stability in future experiments.

The objective is to orient the spacecraft such that its body-fixed x-axis aligns with the target direction:

$$\text{Goal: } \theta_e = \arccos(\mathbf{b}_x \cdot \mathbf{t}) < \theta_{tol} \tag{8}$$

where:

- $\mathbf{b}_x$ is the body x-axis expressed in the inertial frame
- $\theta_{tol} = 0.2$ rad (≈ 11.5) is the tolerance
- Time limit: 200 steps (20 seconds with $\Delta t = 0.1$ s)

The key characteristics of this scenario are the partial observability and the memory requirement. The agent cannot directly observe its orientation, only angular velocity and target direction. Because success requires integrating angular velocity over time to estimate current attitude, the expectation should be that it will favor recurrent policies. The current environment isolates rotational dynamics to focus purely on the effects of partial observability. Incorporating translational dynamics, fuel constraints, or multi-body perturbations is a logical extension that aligns with planned work on full orbital maneuvering scenarios.

This environment serves as a simplified model for several real spacecraft control problems:

1. Earth Observation Satellites: Pointing cameras or sensors at specific ground targets
2. Communication Satellites: Maintaining antenna pointing toward ground stations
3. Space Telescopes: Precision pointing for astronomical observations
4. Solar Panel Orientation: Tracking the sun for optimal power generation
5. Rendezvous and Docking: Attitude alignment during spacecraft proximity operations

The partial observability aspect mirrors real scenarios such as: star trackers being unavailable (e.g., due to sun interference, Earth occlusion), only rate gyroscopes functioning (common failure mode), or low-cost missions with minimal sensor suites (e.g., not all state variables can be observed).

Configuration 1: Full Observability

This configuration represents an idealized scenario where the agent has access to a direct, real-time measurement of its performance. The observation $\mathbf{o}_t$ provided to the agent at time t is a 7-dimensional vector:

$$\mathbf{o}_t = [\underbrace{\omega_x, \omega_y, \omega_z}_{\boldsymbol{\omega}_t}, \underbrace{d_x, d_y, d_z}_{\mathbf{d}_{\text{target}}}, \underbrace{e_p}_{\text{pointing error}}] \tag{9}$$

where:

- $\boldsymbol{\omega}_t \in \mathbb{R}^3$ is the spacecraft's angular velocity in its body frame.
- $\mathbf{d}_{\text{target}} \in \mathbb{R}^3$ is the unit vector of the target direction in the inertial frame.
- $e_p \in [0, \pi]$ is the scalar pointing error, which is the angle between the spacecraft's current pointing direction and the target direction.

Because the pointing error e_p is provided directly, the observation is Markovian. An agent has all the information it needs within a single observation to select an optimal action. It knows its rate of rotation ($\boldsymbol{\omega}_t$) and how far it is from the goal (e_p). A standard, memory-less agent is well-suited for this task.

Configuration 2: Partial Observability

This configuration simulates a more realistic and challenging scenario where a direct measurement of the pointing error is unavailable. The observation $\mathbf{o}_t$ is now a 6-dimensional vector, with the pointing error explicitly removed:

$$\mathbf{o}_t = [\underbrace{\omega_x, \omega_y, \omega_z}_{\boldsymbol{\omega}_t}, \underbrace{d_x, d_y, d_z}_{\mathbf{d}_{\text{target}}}] \tag{10}$$

Without the pointing error, the agent cannot know its current attitude from a single observation. It knows how fast it is rotating ($\boldsymbol{\omega}_t$) but not which way it is currently facing. To solve the task, the agent can infer its orientation by integrating the history of its angular velocities over time. This transforms the problem into a Partially Observable Markov Decision Process (POMDP). A memory-less PPO agent will struggle significantly because two situations

with identical observations could have drastically different attitudes and require opposite control actions.

A recurrent agent, such as PPO-LSTM, is designed for such problems. The LSTM's internal hidden state, $\mathbf{h}_t$, can learn to function as a state estimator. By processing the sequence of observations, the LSTM can effectively maintain an internal representation that approximates the spacecraft's true, unobserved attitude, allowing it to make temporally coherent and informed decisions.

The key difference between the two configurations is whether the attitude error (the difference between current and target orientation) is included in the observation space. The partially observable setting favors policies with memory, such as those used by PPO-LSTM. The LSTM allows the agent to integrate information over time, reconstructing latent state variables (like attitude error) by remembering previous observations and actions.

EXPERIMENTS AND RESULTS

The training environments both leverage BSK-RL Stephenson and Schaub (2024). They are described in detail in the previous section. The only difference between them is the inclusion of the pointing error as an extra observation for the agent in the fully observable case.

Both base algorithm implementations come out of the Stable Baselines 3 project (Raffin et al., 2021). The PPO implementation comes from the main stable-baselines3 repository[1], while the PPO-LSTM uses the RecurrentPPO implementation from the prototype repository, sb3-contrib[2]. The parameters used for both environments was the same between the two environments, and they're provided in the tables below.

The hyperparameters for both PPO and RecurrentPPO, detailed in Tables 1 and 2, were chosen based on a combination of established practices and empirical tuning. Initial values were sourced from the default configurations in the Stable-Baselines3 library, which are benchmarked across a wide range of environments. These were then fine-tuned to ensure stable learning and convergence for our specific attitude control task. The learning rate of 1×10^{-4}, for example, was found to provide a good balance between training speed and stability for both agents.

Table 1: PPO Hyperparameters for Attitude Control

Parameter	**Value**
Policy	MlpPolicy
Learning Rate	1×10^{-4}
n_steps	64
Batch Size	256
n_epochs	10
Entropy Coefficient	0.02
Value Function Coef	0.5
Max Grad Norm	0.5
Network Architecture	[256, 256]
Observation Normalization	True
Reward Normalization	True

Each agent was trained for 200,000 steps, and then evaluated for 100 episodes. Because each episode has a maximum step length of 200 steps (many will terminate sooner), this means that each agent is trained on at least 1,000 episodes. We chose average reward for simplicity and consistency with Stable-Baselines3 evaluation protocols. In the future, including success rate, episode length, and action smoothness would yield additional actionable insights.

In the fully observable environment, the two agents perform very similarly as can be seen in Figure 1. The plots represent a moving average of the raw reward which is aggregated over a window size of 50 episodes. Since training

[1]https://github.com/DLR-RM/stable-baselines3
[2]https://github.com/Stable-Baselines-Team/stable-baselines3-contrib

Table 2: RecurrentPPO Hyperparameters for Attitude Control

Parameter	**Value**
Policy	MlpLstmPolicy
Learning Rate	1×10^{-4}
n_steps	256
Batch Size	64
n_epochs	10
Entropy Coefficient	0.02
Value Function Coef	0.5
Max Grad Norm	0.5
LSTM Hidden Size	256
Network Architecture	[256, 256]
Enable Critic LSTM	True
Observation Normalization	True
Reward Normalization	True

is based on the number of steps conducted, the number of completed episodes is variable.

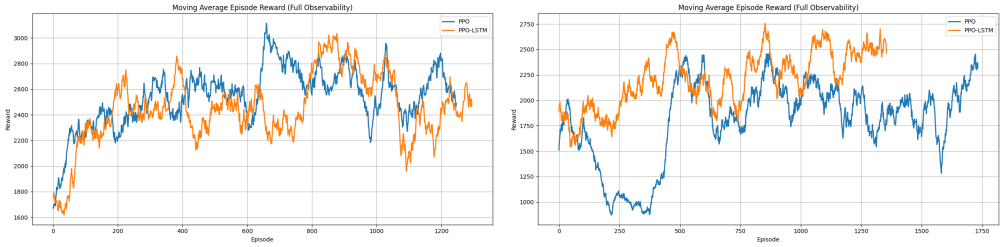

Figure 1: Moving Average of Training Rewards in the Fully Observable Environment

The next plots, shown in Figure 2, show the same moving average reward plot for the partially observable environment described in Configuration 2. In this case, the PPO-LSTM defined by the RecurrentPPO from Stable-Baselines3 has a clear advantage.

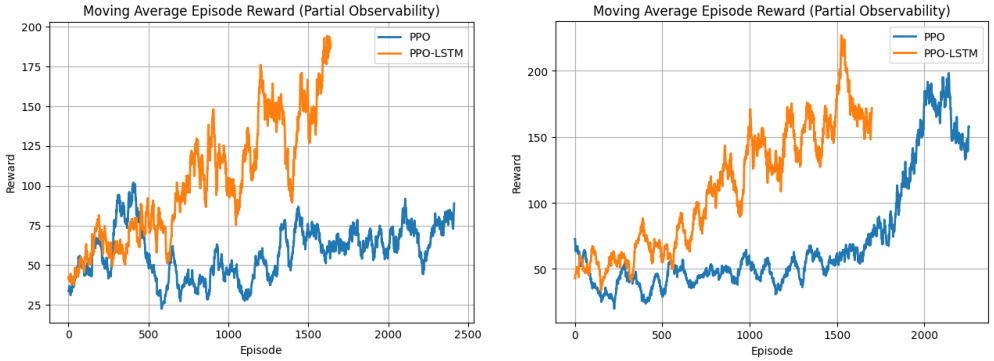

Figure 2: Moving Average of Training Rewards in the Partially Observable Environment

Finally, Figure 3 shows the performance of the learned policy on the evaluation environment. The evaluation environment used here is an exact replica of the training environment, but the performance on the training environment does not always translate perfectly. PPO actually outperformed PPO-LSTM in one of the tests, but performance tends to be very comparable between the two methods. In the partially observable case, PPO-LSTM performs significantly better. In the first trial, it improves the average reward across the evaluation cases by 66.64% and in the second trial it

provided an improvement of 87.69%. To provide a more rigorous comparison, the mean and standard deviation of the average reward across the 100 evaluation episodes are presented in **??**.

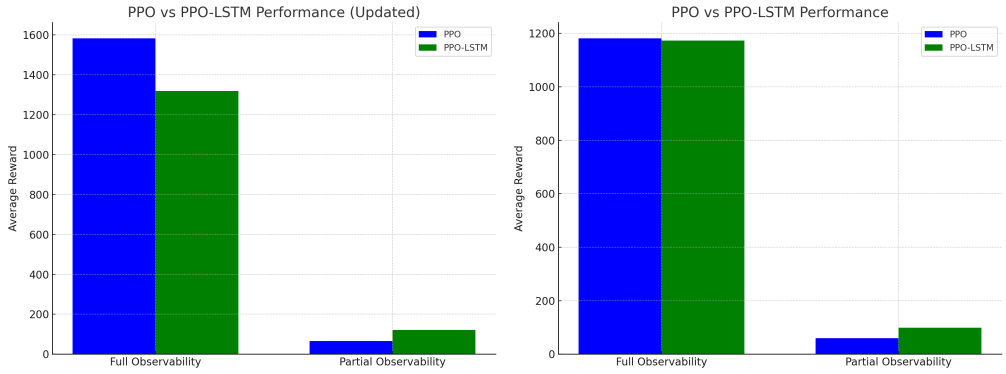

Figure 3: Evaluation rewards comparing PPO and PPO-LSTM

CONCLUSION & FUTURE WORK

Reinforcement learning is proving to be a valuable tool for spacecraft optimization problems. Our findings specifically demonstrate that for complex tasks like attitude control under partial observability—a situation analogous to real-world scenarios like temporary star tracker failure or missions with minimal sensor suites—agents with recurrent memory are not just beneficial, but potentially essential. This has direct implications for developing more robust systems for onboard mission planning, real-time anomaly response, and autonomous rendezvous and docking, where complete state information is often not guaranteed.

While this study focused on the direct comparison between a memory-less policy and a recurrent one, future work should include other techniques for handling partial observability, such as frame stacking. Comparing PPO-LSTM against a PPO agent with stacked observations would help determine if the learned, dynamic memory of an LSTM offers advantages over a fixed, finite history of observations. We limited our initial scope to PPO and PPO-LSTM to ensure a controlled comparison between memory-less and memory-augmented policies under partial observability. Including TD3 or SAC variants is a compelling next step, especially given their deterministic nature and suitability for continuous control. Additionally, exploring ideas designed to improve the robustness of reinforcement learning techniques in the domain of spacecraft optimization (e.g., the varied initialization conditions proposed by Kolosa (2019)) may also prove helpful in the context of mitigating the challenges presented by partially observable environments.

The agents in this study were trained for a relatively short 200,000 steps. While this was sufficient to demonstrate the performance difference in a short-duration mission context, exploring longer training horizons is a key area for future research. Extended training could reveal more about the asymptotic performance of each agent and their robustness over mission timelines that span days or weeks, which is especially relevant for tasks like long-term autonomous orbit maintenance.

REFERENCES

Bakker, B. (2001). Reinforcement learning with long short-term memory. *Advances in neural information processing systems*, 14.

Duell, S., Udluft, S., and Sterzing, V. (2012). Solving partially observable reinforcement learning problems with recurrent neural networks. In *Neural Networks: Tricks of the Trade: Second Edition*, pages 709–733. Springer.

Elkins, J. G., Sood, R., and Rumpf, C. (2020). Autonomous spacecraft attitude control using deep reinforcement learning. In *International Astronautical Congress*.

Forestieri, A. and Casalino, L. (2025). Space trajectory planning with a general reinforcement-learning algorithm. *Aerospace*, 12(4):352.

Gaudet, B., Linares, R., and Furfaro, R. (2020a). Adaptive guidance and integrated navigation with reinforcement meta-learning. *Acta Astronautica*, 169:180–190.

Gaudet, B., Linares, R., and Furfaro, R. (2020b). Six degree-of-freedom body-fixed hovering over unmapped asteroids via lidar altimetry and reinforcement meta-learning. *Acta Astronautica*, 172:90–99.

Gaudet, B., Linares, R., and Furfaro, R. (2020c). Terminal adaptive guidance via reinforcement meta-learning: Applications to autonomous asteroid close-proximity operations. *Acta Astronautica*, 171:1–13.

Herrmann, A. and Schaub, H. (2023). A comparative analysis of reinforcement learning algorithms for earth-observing satellite scheduling. *Frontiers in Space Technologies*, 4:1263489.

Hovell, K. and Ulrich, S. (2021). Deep reinforcement learning for spacecraft proximity operations guidance. *Journal of spacecraft and rockets*, 58(2):254–264.

Kenneally, P. W., Piggott, S., and Schaub, H. (2020). Basilisk: A flexible, scalable and modular astrodynamics simulation framework. *Journal of aerospace information systems*, 17(9):496–507.

Kolosa, D. S. (2019). *A reinforcement learning approach to spacecraft trajectory optimization*. Western Michigan University.

Loettgen, J. L., Ceriotti, M., Aragon-Camarasa, G., and Worrall, K. (2023). Deep reinforcement learning for spacecraft attitude tracking manoeuvres. *University of Glasgow Technical Report*.

Meng, L., Gorbet, R., and Kulić, D. (2021). Memory-based deep reinforcement learning for pomdps. In *2021 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 5619–5626. IEEE.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. (2021). Stable-baselines3: Reliable reinforcement learning implementations. *Journal of machine learning research*, 22(268):1–8.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Scorsoglio, A., D'Ambrosio, A., Ghilardi, L., Gaudet, B., Curti, F., and Furfaro, R. (2022). Image-based deep reinforcement meta-learning for autonomous lunar landing. *Journal of Spacecraft and Rockets*, 59(1):153–165.

Shi, Z., Zhao, F., Wang, X., and Jin, Z. (2022). Satellite attitude tracking control of moving targets combining deep reinforcement learning and predefined-time stability considering energy optimization. *Advances in Space Research*, 69(5):2182–2196.

Stephenson, M., Mantovani, L., Phillips, S., and Schaub, H. (2024). Using enhanced simulation environments to accelerate reinforcement learning for long-duration satellite autonomy. In *AIAA SCITECH 2024 Forum*, page 0990.

Stephenson, M. A. and Schaub, H. (2024). Bsk-rl: Modular, high-fidelity reinforcement learning environments for spacecraft tasking. In *75th International Astronautical Congress, Milan, Italy, IAF*.

Towers, M., Kwiatkowski, A., Terry, J., Balis, J. U., De Cola, G., Deleu, T., Goulao, M., Kallinteris, A., Krimmel, M., KG, A., et al. (2024). Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*.

Wang, X., Deng, Y., Cai, Y., and Jiang, H. (2024). Deep recurrent reinforcement learning for intercept guidance law under partial observability. *Applied Artificial Intelligence*, 38(1):2355023.

Yan, M., Yang, R., Zhang, Y., Yue, L., and Hu, D. (2022). A hierarchical reinforcement learning method for missile evasion and guidance. *Scientific Reports*, 12(1):18888.

Yang, Z., Wu, Z., Wang, Y., and Wu, H. (2024). Deep reinforcement learning lane-changing decision algorithm for intelligent vehicles combining lstm trajectory prediction. *World Electric Vehicle Journal*, 15(4):173.