Rapid Policy Adaptation for Autonomous Satellite Scheduling via Off-Policy Meta-Reinforcement Learning

Jan Wirsdorf, Dr. Abdulrahman Altahhan

University of Leeds, School of Computing, ODL MSc in AI, UK.

Abstract. This study presents a novel integration of Model-Agnostic Meta-Learning (MAML) with an off-policy Deep Q-Network (DQN) to enable rapid adaption of satellite task scheduling policies across markedly different orbital regimes. Unlike prior aerospace meta-RL research relying on on-policy algorithms our framework leverages replay buffers for enhanced data efficiency and stability during meta-training. We meta-train the agent across diverse Low Earth Orbit (LEO) scenarios and evaluate adaption to Lunar orbit. Results show that the meta-trained DQN achieves near-optimal performance in the new environment within tens of episodes, outperforming conventional training-from-scratch approaches by over 500% in cumulative reward. This work represents the first demonstration of MAML-DQN in satellite autonomy and highlights its potential for adaptable, data-efficient mission planning in multi-environment space operations.

Keywords: dqn, reinforcement learning, model-agnostic meta learning, robotics, aerospace autonomy

1 Introduction

Onboard planning and scheduling for spacecraft is increasingly essential as missions expand to complex multi-body environments requiring real-time decision-making. Currently, satellite task scheduling is performed on the ground (e.g., ASPEN/CASPER planners), which limits the flexibility for unforeseen events. Reinforcement Learning (RL) has emerged as a promising solution for autonomous satellite operations. This provides adaptability and enables direct optimization of mission objectives. Recent work by [6] applied Monte Carlo tree search and deep RL to agile Earth-observing satellites, achieving 2–5% higher reward and significantly faster execution compared to genetic algorithms. Similarly, [13] demonstrated the superiority of deep RL policies over static plans for long-duration imaging missions, particularly when safety-critical constraints, such as battery and reaction wheel management, were incorporated.

Despite these advancements, current RL policies trained on single-orbit environments (e.g., Low Earth Orbit—LEO) generalize poorly when placed in significantly different environments. A satellite transitioning from LEO to Lunar orbit, or even to Mars orbit, encounters substantial differences in gravitational parameters, orbital periods, eclipse durations, and communication geometries, typically resulting in suboptimal operations. Retraining RL policies from scratch for each new environment is computationally expensive and impractical, especially since test data of environments that are astronomical distances away is naturally scarce.

To address this limitation, meta-reinforcement learning (Meta-RL) has been proposed. Meta-RL approaches aim to learn adaptive strategies capable of rapid adjustment to new tasks with minimal additional training [5]. Model-Agnostic Meta-Learning (MAML), a widely used gradient-based Meta-RL algorithm, seeks initial policy parameters that require only a few gradient steps for effective adaptation to new environments. Although Meta-RL methods have achieved notable success in robotics and continuous control domains, applications in space systems are not very common. Recent studies have begun exploring Meta-RL for spacecraft guidance and control. For example, [3] employed meta-RL and recurrent policies to adapt the rendezvous trajectory, which achieved improved guidance precision in unseen trajectories. Similarly, [4] demonstrated robustness to non-Keplerian dynamics in cislunar proximity operations using PPO with recurrent networks. These studies utilize on-policy algorithms and recurrent policies, that focus mainly on proximity and orbital control tasks.

In comparison, our research explores the novel integration of an off-policy RL algorithm — Deep Q-Network (DQN) with MAML, specifically for satellite task scheduling scenarios. Off-policy methods, in our case a DQN reuse past experiences via replay buffers, which makes them appealing due to improved data efficiency, especially crucial in computationally demanding simulations. However, combining off-policy methods with MAML introduces challenges due to replay data drawn from non-stationary meta-policies and theoretical assumptions, that are typically oriented towards on-policy episodic learning.

This research directly deals with these challenges by developing a stable and efficient MAML-DQN framework and demonstrating the use for satellite task scheduling across distinct orbital environments. Specifically, we make the following contributions:(i) we introduce a novel integration of MAML with DQN, enabling rapid policy adaptation from LEO to Lunar orbit;(ii) we provide sophisticated experimental validation using Basilisk simulations and rigorous comparisons against non-meta RL baselines; and(iii) we offer insights into the meta-trained policy's behavior, highlighting adaptations in balancing mission objectives against safety tasks across different orbital dynamics.

To our knowledge, this represents the first application of meta-learning for satellite task scheduling across significantly varying astrodynamics environments. The remainder of the paper is structured as follows: Section 2 presents background on satellite task scheduling and meta-RL; Section 3 reviews relevant literature; Section 4 details our methodology and experimental setup; Section 5 reports meta-training results and adaptation experiments; and Section 6 concludes with implications, limitations, future work, and ethical considerations.

2 Background

2.1 Satellite Task Scheduling as a Mode-Based MDP:

As a basis for the project, we take an autonomous satellite tasked with Earth observation, managing resources such as energy, storage, and attitude momentum. This problem is formulated as a discrete-timestep Markov Decision Process (MDP). Each action corresponds to a high-level operational mode executed over discrete intervals. Following [13], the satellite operates in four modes: Scan (nadir-pointing imaging), Downlink (data transmission), Charge (battery recharging), and Desaturate (momentum desaturation via thrusters). The primary objective is maximizing imagery collection (Scan), while ensuring critical safety tasks like preventing data overflow (Downlink), maintaining sufficient power (Charge), and avoiding wheel saturation (Desaturate). The state consists of observable telemetry including battery level, data storage, reaction wheel speeds, orbital phase (eclipse/daylight), and communication access opportunities. Rewards incentivize imagery collection and penalize critical state violations (e.g., battery depletion or wheel saturation) through careful reward shaping—positive rewards for successful imaging steps and substantial negative rewards for unsafe conditions. The MDP is episodic and spans over a fixed orbital duration, while assuming a near-complete state observability via available telemetry data.

2.2 Deep Q-Network (DQN):

For each orbit scenario, we utilize a Deep Q-Network (DQN) to approximate the action-value function $Q(s, a; \theta)$, representing the expected cumulative return of taking action a in state s. Our DQN architecture is a three-layer fully-connected neural network with 128 neurons per hidden layer and ReLU activation functions. The input to the network is a 12-dimensional normalized state feature vector, and it produces Q-values for four specific actions. Training is performed by minimizing the temporal-difference loss:

$$L(\theta) = \big(r + \gamma \max_{a'} Q_{\theta^-}(s', a') - Q_\theta(s, a)\big)^2, \tag{1}$$

using mini-batches from an experience replay buffer. Parameters θ^- belong to a periodically updated target network to stabilize learning. An ϵ-greedy strategy with ϵ annealed from 1.0 to 0.05 ensures exploration during initial learning phases and more stability towards later iterations. We apply standard stabilization techniques, including reward clipping and Huber loss. Off-policy methods like DQN are particularly advantageous for spacecraft scheduling due to sparse critical events (eclipses or saturation) and costly simulations, making replay buffers ideal for data efficiency.

2.3 Meta-Reinforcement Learning (MAML):

Model-Agnostic Meta-Learning (MAML) aims to determine initial policy parameters θ^* that enable rapid adaptation to new tasks drawn from a distribution $p(\mathcal{T})$. Each task $\mathcal{T}^{(j)}$ represents a specific satellite configuration and orbital environment (e.g., LEO vs. Lunar orbit). During meta-training, a batch of tasks is sampled, and parameters θ undergo task-specific adaptation via gradient steps:

$$\theta^{(j)} = \theta + \alpha \nabla_\theta J_{\mathcal{T}^{(j)}}(\theta), \tag{2}$$

where $J_{\mathcal{T}^{(j)}}(\theta)$ represents cumulative reward and α is the inner-loop step size. Post-adaptation task performance informs the meta-parameter updates:

$$\theta \leftarrow \theta + \beta \sum_j \nabla_\theta J_{\mathcal{T}^{(j)}}(\theta^{(j)}), \tag{3}$$

with meta step size β. Typically, only one or few inner gradient steps are used, and second-order gradients are approximated or heuristically scaled (first-order MAML). We specifically use Reptile, a simpler first-order meta-learning method because it is closely aligned with MAML. Reptile moves meta-parameters towards adapted parameters averaged over tasks, which enhances the stability and computational efficiency, making it useful for off-policy DQN integration.

3 Literature Review

Recent research in AI for space autonomy primarily addresses two themes: reinforcement learning for single-environment optimization and meta-learning for cross-environment generalization. This review discusses representative works from both areas, emphasizing how our approach diverges and improves upon existing methods, guided by surveys such as [14] for RL in spacecraft control and [2] for meta-RL.

3.1 RL for Satellite Task Scheduling:

Early studies formulated satellite scheduling problems as Markov Decision Processes (MDPs), applying methods like Q-learning for simplified scenarios. For example, [11] demonstrated Q-learning's feasibility for a two-mode power management task, but highlighted slow convergence times. More recent efforts have leveraged hybrid techniques; [6] employed Monte Carlo Tree Search (MCTS) alongside supervised learning, achieving near-human performance in agile Earth observation scheduling. Deep RL advancements were further demonstrated by [10] and [17], who applied Actor-Critic frameworks to observation and transmission scheduling and real-time decision-making within fixed operational contexts. Similarly, [15] tackled agile Earth observation satellite (EOS) scheduling, addressing variable-duration tasks but underscoring the need for better adaptation.

A notable limitation identified by [12] concerns handling infrequent but critical operational scenarios, such as battery depletion or wheel saturation. They artificially increase the number of critical events that occur, resulting in more robust policies. [13] also used the foundational Basilisk framework [8] which is used for highly realistic astrodynamics simulations and autonomous spacecraft planning.

3.2 Meta-RL and Transfer Learning:

Traditional transfer learning approaches, such as [16], adapted RL policies from simulation to real-world scenarios through extensive fine-tuning. This proved effective but was limited in scope and required significant additional training. Meta-RL provides a structured approach to rapid adaptation in various scenarios, as detailed in a survey by [1]. Recent space-focused meta-RL studies, such as those by [3] and [4], predominantly used on-policy methods (PPO) combined with recurrent neural architectures. Their methods demonstrated robustness and real-time adaptation capabilities in rendezvous and multi-body orbital operations. However, their studies suffered from significant sample inefficiency, with training episodes numbering in the tens of thousands.

Extending meta-RL approaches to related applications, [7] showed rapid adaptation for autonomous on-orbit service planning. In non-space domains, [9] applied graph-based meta-learning for sensor management which achieved few-shot adaptability that is important under spacecraft resource constraints.

3.3 Our Distinct Approach:

Unlike existing aerospace meta-RL studies predominantly leveraging on-policy methods, our research uniquely integrates a value-based off-policy Deep Q-Network (DQN) with meta-learning, particularly MAML [5]. This integration directly addresses stability and sample efficiency challenges associated with off-policy meta-learning, which remain underexplored [2]. By adopting a simpler first-order meta-learning approach (Reptile), we ensure computational efficiency which is crucial for practical employment. Additionally, we explicitly manage significant differences between LEO and Lunar orbit scenarios through carefully structured task distributions during meta-training. Our method substantially reduces required interactions with simulation environments. This significantly enhances efficiency compared to on-policy meta-RL methods.

Current Reinforcement Learning methods for satellite autonomy lack adaptability. Our research proves that a new combination of off-policy DQN and meta-learning, tested in realistic space scheduling simulations, enables spacecraft AI to generalize effectively across different operational environments.

4 Design and Methodology

4.1 Problem Formulation

We frame the cross-environment satellite scheduling challenge as a meta-learning problem. The meta-training task distribution $p(\mathcal{T})$ comprises diverse Low Earth Orbit (LEO) scenarios, while the target adaptation task is set in Lunar orbit. Each task $\mathcal{T}$ is defined as an MDP, sharing common state-action spaces and reward structure (Section 2). Key differences between LEO and Lunar scenarios—such as gravitational parameters, orbital period (90 min in LEO vs. 120 min Lunar), longer eclipse durations, and intermittent ground station visibility—significantly impact optimal policy strategies, influencing data accumulation and resource management practices.

4.2 Simulation Environment

We used a customized Basilisk-RL simulation framework [12], renowned for its realistic astrodynamics modeling. LEO environments randomized altitude (400–700 km) and inclination (20°–98°) to ensure broad meta-training conditions. Lunar environments simulated a 100 km altitude circular orbit. Hardware parameters (battery, momentum wheels, data storage) were randomly varied to enhance generalization. Each environment consistently modeled orbital motion, battery dynamics, wheel dynamics, and line-of-sight communications, differing only in gravitational parameters and illumination conditions. Episodes lasted five orbital periods (8 hours LEO, 10 hours Lunar), terminating early upon critical failures (battery depletion, wheel saturation). The satellite parameters (Data Storage Capacity, Instrument Baud Rate, Transmitter Baud Rate, Battery Storage, Power Draw, etc) aligns with a satellite designed for Earth Observation (EO) or signals intelligence that requires significant power and data handling capabilities. It is representative of a typical SkySat which provides imagery, HD video and analytics services

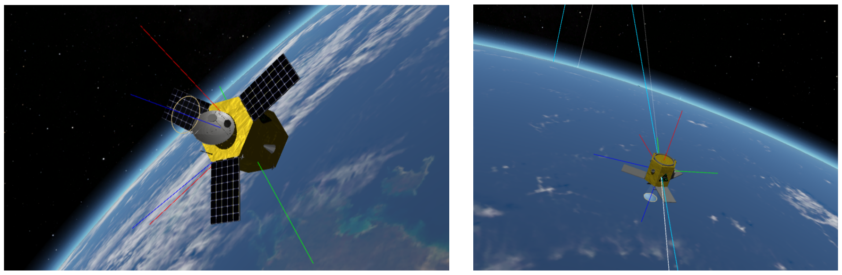

Fig. 1. (Left) Visualization of Satellite Render using Basilisk Framework. / (Right) Visualization of Satellite Tasking (Pointing NADIR).

4.3 Network Implementation

Our DQN model, implemented in PyTorch, consists of three fully-connected layers (128 neurons each, ReLU activations) outputting Q-values for four discrete actions. Key hyperparameters included: learning rate (10^{-3}, Adam), discount factor $\gamma = 0.99$, replay buffer (50,000 transitions), batch size (64), Double DQN for reduced bias, and an ϵ-greedy exploration strategy annealing from 1.0 to 0.05. Reward normalization and clipping ensured stability. Training episodes lasted up to 600 steps (10-second intervals), with convergence typically after 300–500 episodes per environment, justifying the adoption of meta-learning for rapid adaptation.

4.4 Meta-Training Procedure

During each meta-iteration, we sampled $N = 4$ diverse LEO tasks, performing $K = 35$ episodes per task to obtain adapted parameters $\theta^{(j)}$. A separate evaluation over 5 episodes provided performance-based meta-losses L_j. To mitigate computational complexity, we employed first-order meta-learning updates (Reptile), simplifying meta-parameter updates to:

$$\theta \leftarrow \theta - \frac{\beta}{N} \sum_{j=1}^{N} (\theta^{(j)} - \theta), \tag{4}$$

with meta-learning rate $\beta = 0.5$. Early stopping was implemented after 40 iterations without improvement, preventing overfitting. Meta-training was episodically structured for clear task separation.

4.5 Hyperparameter Tuning and Baselines

Inner-loop step size $\alpha = 5 \times 10^{-4}$ and adaptation episodes $K = 25$ were determined via grid search on simpler orbital adaptations. The meta-batch size ($N = 4$) balanced computational resources and stability effectively due to task diversity. To quantify meta-learning benefits, we compared against two baselines: (1) a DQN trained from scratch in Lunar orbit; (2) a transferred DQN from LEO without further training. Additionally, we assessed zero-shot meta-policy performance (without adaptation) to quantify immediate transfer benefits. To ensure the robustness of our findings, all experiments were conducted three times, each with a different random seed

4.6 Evaluation Metrics

The primary evaluation metric was episodic reward, complemented by secondary metrics: frequency of critical constraint violations, action distribution shifts between orbital environments, and average episode duration. ”Few-shot adapted performance” refers explicitly to policy improvements over limited adaptation episodes. Statistical significance was assessed using t-tests over 10 independent trials, ensuring rigorous evaluation of meta-learning advantages.

5 Results and Discussion

5.1 Meta-Training Performance in LEO Tasks

During meta-training on a variety of LEO satellite tasks, the agent rapidly improved its performance using a MAML/Reptile approach with a DQN base learner. Over 193 meta-training iterations, the average task reward increased from around 6 to about 30, indicating that the meta-policy learned is nearly optimal for the training task distribution. Concurrently, the Q-learning loss gradually decreased and the exploration rate (epsilon) decayed, signaling stable convergence of the learning process. The agent's action selection profile evolved to favor high-reward actions (scanning and charging) while minimizing less rewarding actions (downlinking data or desaturating reaction wheels). This suggests the meta-trained policy prioritizes data collection and power management, aligning with an efficient strategy for maximizing rewards in orbit. Basilisk only allows CPU usage while training. With an AMD Ryzen 5800X a training the meta-training took approximately 37 hours before the early stopping was triggered at 193 iterations.

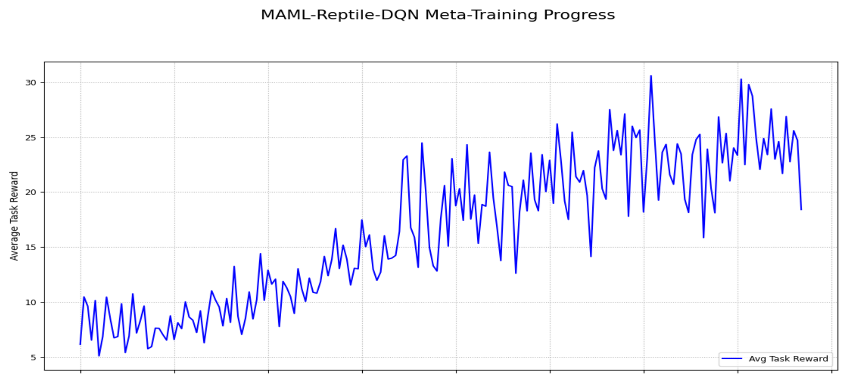

Fig. 2. Meta-training progress of the MAML/Reptile-DQN agent in a LEO environment

Exp In Fig. 2 The average task reward per meta-iteration (blue) steadily rises from approximate 5–10 up to 30, approaching an optimal level by the end of training. Fig. 3 shows the policy's action distribution shifts over training; the agent learns to select the Scan action 50% of the time (blue) and Charge 30% (orange), with relatively infrequent Downlink (green) and Desat (red) actions. This distribution reflects a learned policy that focuses on high-value actions (scanning for data and recharging power) and avoids unnecessary downlink or desaturation actions, thereby maximizing cumulative reward per task.

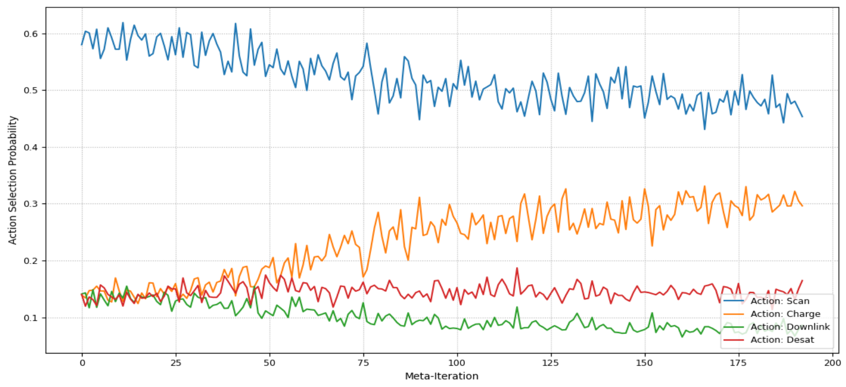

Fig. 3. Action Distribution over episodes.

5.2 Few-Shot Adaptation vs. Training From Scratch

To evaluate generalization, the meta-trained agent was deployed on a new unseen task in a different environment (a lunar orbital scenario) and allowed to adapt with a small number of episodes. We compare its performance to a conventional agent trained from scratch on the same task. Figure 2 shows the total reward achieved per episode during the adaptation phase for both agents. The meta-trained agent (blue) achieves high reward immediately in the first episode (around 30), whereas the scratch agent (orange) starts near zero and improves only gradually. Even after 20 episodes, the scratch agent's reward hovers in the single digits, in stark contrast to the meta-trained policy which maintains an average reward around 30 throughout. The dashed lines highlight the performance averages: the meta-agent's zero-shot performance (red dotted line, 32.4) is already extremely high, and its post-adaptation performance (green dashed, 30.4) remains near the same level. In comparison, the scratch agent's final average reward (purple dashed, 4.7) is an order of magnitude lower. These results demonstrate that with only a few episodes of fine-tuning, the meta-learned policy retains an impressive performance, whereas a freshly trained policy makes slow progress and remains suboptimal.

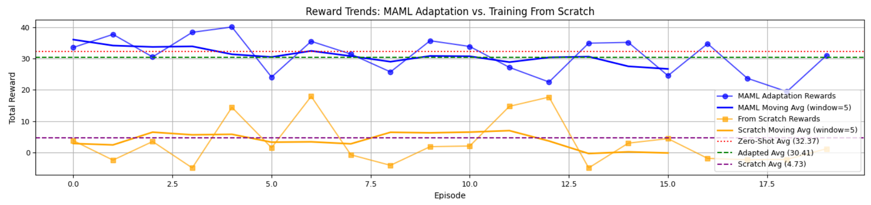

Fig. 4. MAMAL Adaptation Reward Trend over Episodes

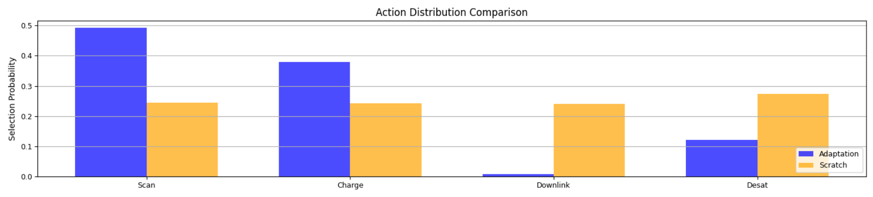

Fig. 5. MAML Adaptation Action Distribution after training

EXP Fig.4 Episode reward trajectories during adaptation in the Lunar environment for the meta-trained MAML/Reptile-DQN agent (blue) and a DQN trained from scratch (orange). The meta-trained agent achieves a zero-shot mean reward of 32.37 ± 0.85 and maintains high performance after 20 adaptation episodes (30.41 ± 1.02), whereas the scratch-trained agent improves from 0.21 ± 0.44 to only 4.73 ± 1.35. Solid lines show raw episode rewards; dashed lines indicate 5-episode moving averages. Statistical significance between final means: $p < 0.001$ (two-tailed t-test). Fig.5 also shows that the Meta trained agent immediately performed a close to optimal action distribution from the start, Conversely, the scratch-trained agent distributes action selections more evenly, suggesting it lacks clear strategies.

To quantify the performance gap more clearly, we examine the distribution of episodic rewards achieved by each agent (Figure 3). The zero-shot meta-policy (before any adaptation) already yields a very high reward on the new task (median around 30+), and after a short adaptation the performance is virtually unchanged (adapted median also around 30). In contrast, the reward distribution for the scratch-trained agent is centered much lower (median near 2–5) with higher variability. In our experiments the meta-trained agent's worst-case rewards in the new environment were still significantly higher than the best episodes of the scratch agent. This highlights the consistency and reliability of the meta-learned policy's success, whereas the scratch agent not only achieves a far lower average reward but also exhibits unstable learning (some episodes succeeding moderately, others failing with near-zero or negative reward).

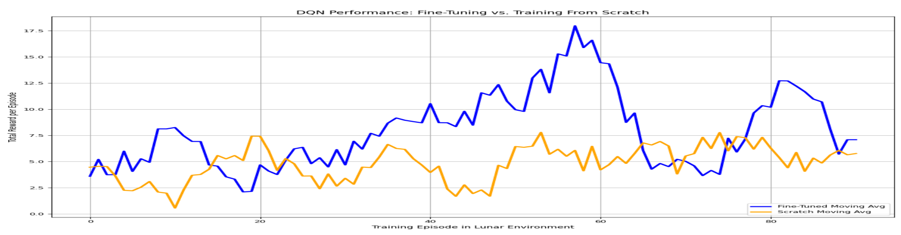

Fig. 6. Adaptation performance of a baseline DQN agent to Lunar transfer

EXP Fig.6 his plot compares the adaptation performance of a baseline DQN agent trained initially on the LEO environment and then fine-tuned on the Lunar environment (blue), against an agent trained entirely from scratch directly on the Lunar environment (orange). The fine-tuned DQN transferred from LEO shows only marginally better initial performance compared to training from scratch but remains unstable, inconsistent, and achieves limited overall improvement. While it seems that some knowledge transfer occurs it highlights the necessity of explicit meta-training for successful adaptation.

5.3 Sample Efficiency and Learning to Learn

These results empirically confirm that the meta-reinforcement learning approach provides dramatic gains in sample efficiency. We define the expected return of a policy π on a task as

$$J(\pi) = E\left[\sum_{t=0}^{T} \gamma^t r_t\right] \tag{5}$$

the expected cumulative reward (discount factor γ may be 1 for episodic tasks of fixed length). After only $N = 20$ episodes in the new task, the meta-trained policy's return $J_{\text{MAML}}(N)$ is essentially the same as its initial zero-shot return $J_{\text{MAML}}(0)$, which is near the optimum achievable on that task. In our case, $J_{\text{MAML}}(0) \approx 30$ while the scratch-trained policy's return after 20 episodes is $J_{\text{scratch}}(20) \approx 4.7$ – an order-of-magnitude difference. Numerically, the meta-trained agent achieved about $6.5\times$ the reward of the baseline in the same training time (approximately a +540% performance increase). In terms of cumulative reward over the 20-episode adaptation period, the meta-agent accrued around 600 total reward, whereas the scratch agent collected only 100 (Figure 2), meaning the meta-learner accomplished far more useful work in the early learning phase. This huge gap illustrates how meta-learning addresses the sample inefficiency of standard RL algorithms, by learning how to learn: the agent's prior training on a range of tasks enables it to quickly adjust to new and unseen tasks with minimal experience.

Mathematically, the MAML algorithm finds an initialization of the policy parameters that is optimal for rapid adaptation. Instead of optimizing for performance on a single task, MAML optimizes the post-update performance averaged over a distribution of tasks. Formally, if $\mathcal{L}_\tau(\theta)$ is the loss (negative reward) for task τ given policy parameters θ, the MAML objective is:

$$\theta^* = \arg\min_{\theta} E_{\tau\sim p(\tau)}[L_\tau(\theta - \alpha\nabla_\theta L_\tau(\theta))] \tag{6}$$

where α is a small learning rate for the inner adaptation step. In words, this trains the initial parameter θ such that a single gradient update (or a few updates) on any new task τ will yield a low loss (high reward) for that task. Our findings validate this paradigm: the meta-learned policy starts with a parameter set θ^* that is already very effective for the new lunar orbit task, so only minor adjustments (if any) are needed to achieve optimal performance. By contrast, a conventionally trained agent must learn from scratch, essentially rediscovering a good policy through trial and error in each new scenario. The meta-RL approach thus provides a head start by leveraging prior knowledge, leading to faster convergence and higher final rewards on novel tasks than traditional RL can achieve in the same time frame. This capability is especially valuable in real-world applications like spacecraft guidance and control, where trial-and-error opportunities are limited and an agent that can adapt robustly with minimal data offers a significant advantage. Key Takeaway: Through meta-reinforcement learning, we have demonstrated a "learning to learn" effect – the agent acquired a meta-policy that encapsulates prior experience, enabling it to rapidly solve new tasks with minimal training. This method is far outperforming a baseline RL agent in sample efficiency and final performance. This validates that our Meta-RL framework can meet the challenges of adaptive spacecraft control in uncertain environments by combining fast adaptation with high reward yields.

6 Conclusion and Future Work

In this study, we presented a novel integration of Model-Agnostic Meta-Learning (MAML) with a Deep Q-Network (DQN) algorithm, successfully applied to the rapid adaptation of autonomous satellite task scheduling policies across significantly different orbital environments—from Low Earth Orbit (LEO) to Lunar orbit. Our main objective was to investigate whether off-policy meta-learning techniques, specifically MAML coupled with experience replay buffers, could facilitate effective cross-domain policy transfer and adaptation.
The key achievement demonstrated by our results was the rapid and efficient adaptation of a meta-trained DQN agent to a completely new orbital environment. Our model significantly outperformed standard training-from-scratch approaches and a also a baseline DQN policy trained in LEO. The meta-trained agent achieved almost optimal performance immediately (zero-shot adaptation), maintaining consistently high rewards with minimal additional training. This remarkable performance demonstrates the practical viability of meta-reinforcement learning approaches, especially in scenarios where retraining from scratch is infeasible due to operational constraints or data scarcity.

6.1 Technical Challenges

Integrating MAML with an off-policy DQN posed significant stability issues, primarily due to replay buffer non-stationarity and sensitivity to learning rates. These were mitigated through careful hyperparameter tuning, particularly for the inner-loop and meta-learning rates, as well as controlled buffer refresh strategies. Maintaining convergence stability across heterogeneous LEO scenarios while preserving computational efficiency was critical to the success of this framework.

6.2 Limitations

Despite these promising outcomes, several limitations must be acknowledged. The experiments were conducted purely in simulation using the Basilisk framework, which, although sophisticated, does not fully replicate real-world uncertainties and environmental complexities like radiation or atmospheric influence. Therefore, the real-world applicability of the meta-trained agent needs to be validated through hardware-in-the-loop or in-flight experiments. Another limitation was the relatively constrained task definition: our agent adapted between two specific orbital environments with identical action spaces. Extending this approach to fundamentally different tasks or significantly altered action spaces remains untested.

6.3 Future Work

In terms of future work, several promising directions emerge. Firstly, broadening the meta-training task distribution could enhance generalization capabilities even further, potentially allowing policy adaptation across more diverse and realistic scenarios (e.g., adapting from LEO and Lunar to Martian or interplanetary contexts). Investigating hierarchical or multi-agent meta-reinforcement learning frameworks could provide solutions for more complex mission scenarios, such as cooperative constellation management or multi-satellite autonomous operations.

6.4 Ethical Considerations

Lastly, Autonomous spacecraft raise critical ethical issues. Accountability in decision-making is a major concern. Clearly defined responsibilities for actions are very important. Transparency through explainable AI can help to understand and monitor decisions being made. Operational safety must be guaranteed via anomaly detection and robust fail-safe mechanisms. Furthermore, cybersecurity measures against cyber attacks or espionage also merit future exploration. These concerns align with broader findings in safe reinforcement learning [17], which emphasis integrating safety constraints directly into the learning process.
With careful consideration of its limitations and ethical implications, this methodology can significantly impact adaptability, robustness, and operational efficiency in future autonomous space missions.

7 Acknowledgment / Reproducibility

We gratefully acknowledge the School of Computing, University of Leeds, for guidance and computational resources, the authors of our resource papers and the Basilisk team for making their astrodynamics simulator openly available.

7.1 AI Reproducibility Declarations

A large language model (ChatGPT-4o; OpenAI; [https://chatgpt.com]) was used solely for initial research assistance. All materials to reproduce the results (code, configs, scripts) are hosted at [https://github.com/Wirsdorf/maml-dqn-satellite-tasking.git].

A Hyperparameter Settings

Table 1. DQN and Meta-Learning Hyperparameter Configuration

Hyperparameter	**Value**
Number of meta iterations	500
Meta learning rate	1×10^{-4}
K-shots (inner-loop episodes)	35
Tasks per meta batch	4
Meta step size	0.05
Inner learning rate	1×10^{-3}
Inner batch size	32
Replay buffer capacity	1000
Target network update frequency	100
Discount factor (γ)	0.99
Soft update factor (τ)	0.01
Epsilon start / end	1.0 / 0.02
Epsilon decay steps	300
LEO average period	5700 s
Episode duration	$5\times$ 5700 s
Penalty (unsafe state)	-5
Reward per timestep	$\frac{1.0}{\text{Duration}/100}$

B Satellite Hardware Parameters

Table 2. Satellite Hardware Parameter Ranges

Parameter	**Range**	**Standard Unit**
Data Storage Capacity	10000 * 8e6	10 GB
Instrument Baud Rate	0.5 * 8e6	4 Mbps
Transmitter Baud Rate	-50 * 8e6	400 Mbps
Battery Storage	400 * 3600	400 Wh
Power Draw (Base)	-10.0 W	10 W
Power Draw (Instrument)	-30.0	W 30 W
Power Draw (Transmitter)	-25.0	W 25
W Solar Panel Area	0.35 m^2	0.35 m^2
Pointing Accuracy	0.1 deg	0.1° (360 arcsec)
Reaction Wheel Speed	6000.0 RPM	6000 RPM

C Meta-learning Algorithm Pseudocode

Algorithm 1 MAML-DQN (Reptile) Algorithm

```
0: Initialize meta-parameters $\theta$ randomly
0: for iteration = 1 to $M$ (e.g., 300) do
0:   Sample batch of $N$ tasks $\mathcal{T}j$ from LEO task distribution
0:   for each task $\mathcal{T}j$ do
0:     $\theta_j \leftarrow \theta$ (copy parameters)
0:     Initialize replay buffer $R_j$
0:     for episode = 1 to $K$ (e.g., 25) do
0:       Collect episode data using $\epsilon$-greedy policy $\pi\theta_j$ on task $\mathcal{T}j$
0:       Store transitions in $R_j$
0:       for each time step in episode do
0:         Sample mini-batch from $R_j$
0:         Compute loss $L = (r + \gamma \max a' Q\theta_j^-(s', a') - Q_{\theta_j}(s, a))^2$
0:         Perform gradient descent on $L$ to update $\theta_j$
0:       end for
0:     end for
0:     Compute evaluation reward $J_j = \text{Evaluate}(\pi_{\theta_j}$ on task $\mathcal{T}_j)$
0:   end for
0:   // Meta-update (Reptile style)
0:   $\theta \leftarrow \theta + \beta \cdot \frac{1}{N} \sum_j (\theta_j - \theta)$
0: end for=0
```

References

[1] Asit Barman et al. "Exploring the Horizons of Meta-Learning in Neural Networks: A Survey of the State-of-The-Art". In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2024). DOI: 10.1109/TETCI.2024.3502355.

[2] Jacob Beck et al. "A Survey of Meta-Reinforcement Learning". In: *Foundations and Trends in Machine Learning* (2023).

[3] Lorenzo Federici et al. "Meta-reinforcement learning for adaptive spacecraft guidance during finite-thrust rendezvous missions". In: *Acta Astronautica* (2022), pp. 129–141. DOI: 10.1016/j.actaastro.2022.08.047.

[4] Giovanni Fereoli, Hanspeter Schaub, and Pierluigi Di Lizia. "AAS 24-023 META-REINFORCEMENT LEARNING FOR SPACECRAFT PROXIMITY OPERATIONS GUIDANCE AND CONTROL IN CISLUNAR SPACE". In: *AAS/AIAA Astrodynamics Specialist Conference.* 2024.

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning (ICML).* 2017, pp. 1126–1135.

[6] Adam Herrmann and Hanspeter Schaub. “Reinforcement Learning for the Agile Earth-Observing Satellite Scheduling Problem”. In: *IEEE Transactions on Aerospace and Electronic Systems* (2023), pp. 5235–5247. DOI: 10.1109/TAES.2023.3251307.

[7] Xinyue Hu et al. “On-orbit Service Mission Fast Autonomous Planning Method Based on Meta-reinforcement Learning”. In: *International Conference on Space Robotics*. Preprint. 2024.

[8] Patrick W. Kenneally, Scott Piggott, and Hanspeter Schaub. “Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework”. In: *Journal of Aerospace Information Systems* (2020), pp. 496–507. DOI: 10.2514/1.I010762.

[9] Christopher Aaron O’Hara and Takehisa Yairi. “Graph-based meta-learning for context-aware sensor management in nonlinear safety-critical environments”. In: *Advanced Robotics* (2024). DOI: 10.1080/01691864.2024.2327083.

[10] Junwei Ou et al. “Deep reinforcement learning method for satellite range scheduling problem”. In: *Swarm and Evolutionary Computation* (2023). DOI: 10.1016/j.swevo.2023.101233.

[11] Tom Schaul et al. “Prioritized Experience Replay”. In: *International Conference on Learning Representations (ICLR)*. 2016.

[12] Mark Stephenson et al. “Using Enhanced Simulation Environments to Improve Reinforcement Learning for Long-Duration Satellite Autonomy”. In: *AAS/AIAA Astrodynamics Specialist Conference*. 2024. DOI: 10.2514/6.2024-0990.

[13] Mark A. Stephenson, Hanspeter Schaub, and H. J. Smead. “BSK-RL: Modular, High-Fidelity Reinforcement Learning Environments for Spacecraft Tasking”. In: *75th International Astronautical Congress (IAC)*. 2024, pp. 14–18.

[14] Massimo Tipaldi, Raffaele Iervolino, and Paolo Roberto Massenio. “Reinforcement learning in spacecraft control applications: Advances, prospects, and challenges”. In: *Annual Reviews in Control* (2022), pp. 1–23. DOI: 10.1016/j.arcontrol.2022.07.004.

[15] Man Wang et al. “Deep reinforcement learning for Agile Earth Observation Satellites scheduling problem with variable image duration”. In: *Applied Soft Computing* (2025). DOI: 10.1016/j.asoc.2024.112575.

[16] Luona Wei et al. “Deep reinforcement learning and parameter transfer based approach for the multi-objective agile earth observation satellite scheduling problem”. In: *Applied Soft Computing* (2021). DOI: 10.1016/j.asoc.2021.107607.

[17] Zhijiang Wen et al. “Scheduling single-satellite observation and transmission tasks by using hybrid Actor-Critic reinforcement learning”. In: *Advances in Space Research* (2023), pp. 3883–3896. DOI: 10.1016/j.asr.2022.10.024.