(Preprint) AAS 26-XXX

# SHIELDING AGAINST THE UNSAFE: EVALUATION OF SHIELDED DEEP REINFORCEMENT LEARNING TRAINING APPROACHES FOR AUTONOMOUS SPACECRAFT

**Lorenzzo Quevedo Mantovani*** **and Hanspeter Schaub†**

Deep reinforcement learning (DRL) approaches have been proposed to solve the agile Earth observing satellite (AEOS) scheduling problem. DRL-based policies demonstrate the potential for real-time on-board decision-making, enabling greater adaptability and performance in dynamic environments. Despite the promising results, safety remains a critical concern, particularly in high-stakes applications such as autonomous spacecraft operations, where human intervention may be limited. Shields have demonstrated minimal performance degradation while providing safety guarantees in the AEOS context. However, integrating shields into the learning process for AEOS remains an active area of research. Therefore, this paper explores two approaches to shielded training: action replacement with interference penalty, and action masking. Testing episodes thirty times longer than training episodes are used to evaluate policies in long-term operations and the trade-off between safety and performance. Results indicate that action replacement methods with interference penalties during training tend to improve safety without significantly compromising performance. Additionally, introducing the interference penalty during training enables agents to learn the safety aspects of the problem more effectively, resulting in fewer safety violations when tested without shields and shield intervention when tested with shields.

## INTRODUCTION

Earth observing satellites (EOS) are equipped with sensing instruments to acquire images of Earth's surface, which can be used for multiple purposes such as crop monitoring, intelligence gathering, and disaster response. During the spacecraft's operation, there is a planning and scheduling phase responsible for providing the sequence of actions the satellite should take to meet the mission requirements while respecting the system's constraints. The advent of agile EOSs (AEOSs) adds complexity to the problem due to their extra maneuverability capabilities, both along- and across-track, increasing the solution space. The growing demand for satellite imagery has led to oversubscribed systems, making the scheduling problem increasingly complex.

The AEOS scheduling problem is shown to be NP-hard, and different optimization techniques are investigated to solve it, including greedy algorithms, dynamic programming, and constraint programming.[1] Heuristics,[2] local search,[3] infeasibility-based graph,[4] budget uncertainty[5,6] have also been explored. Despite many advances, the need for fast replanning and real-time decision-making remains a challenge. Deep reinforcement learning (DRL) techniques are proposed to tackle

*Ph.D. Student, Ann and H.J. Smead Department of Aerospace Engineering Sciences, 3775 Discovery Drive, Boulder, CO 80303, AIAA Student Member.

†Distinguished Professor and Department Chair, Ann and H.J. Smead Department of Aerospace Engineering Sciences, 3775 Discovery Drive, Boulder, CO 80303, AIAA and AAS Fellow.

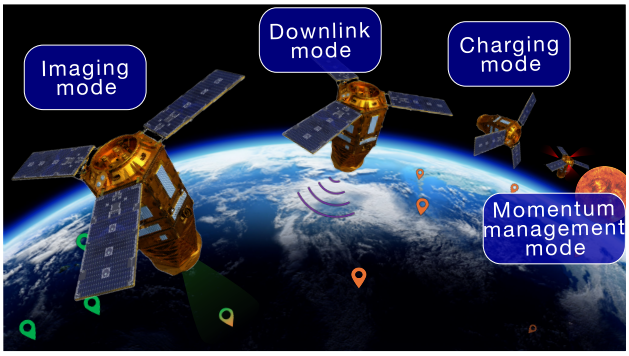



**Figure 1**: Illustration of AEOS running DRL-based policies for on-board closed-loop decision-making.

these issues, accounting for complex dynamics and constraints.[7–11] Figure 1 illustrates AEOSs running DRL-based policies in a closed-loop format.

Despite the advances provided by DRL-based policies, using DRL for on-board autonomy raises safety concerns. Shields can be added to the system to provide a layer of safety guarantees with minimal interference.[12,13] Shields can be incorporated during training and deployment, or exclusively during deployment. The different forms of integrating shields in the training pipeline can result in varying degrees of safety and performance trade-offs during deployment.[14] The three main approaches are action replacement, action masking, and action projection.

In action replacement, the shield will replace the agent's action with a safe action if the original selected action is deemed unsafe. In contrast, action masking only allows the agent to sample from a set of safe actions, effectively filtering out unsafe actions. Lastly, action projection tries to find a safe action while minimizing its distance from the original action, mainly applicable to continuous action spaces. In addition to these three main approaches, the policy can be updated with the corrected safe action or the original action during training. Although existing comparative studies investigate such approaches, there is a need for further research on these different methods in more complex scenarios and how these methods can interact with the policy training.[14] More specifically, the training of DRL-based policies with shields remains underexplored in the AEOS context.

Shields are applied to the satellite scheduling problem when considering a nadir scan action with battery levels and reaction wheel speeds as safety considerations.[15] In a similar environment, training the agent with shields improved safety and performance metrics compared to unshielded training.[16] In the Earth observation case with point targets, the satellite's angular velocity is also included as a safety aspect.[17] In the rendezvous context, control barrier functions are used to create an action projection shield, and it is shown that enforcing some constraints can be beneficial during training.[18] Additionally, the use of optimization techniques with action projection shields is also proposed to keep the spacecraft in a safe trajectory.[19]

The use of shields during the testing phase (after training) is shown to provide safety guarantees for long-term operations without significantly hurting performance.[20] Still, the different training methods with shields remain unexplored in the AEOS context. Therefore, this research aims to fill this gap by investigating the effect of action replacement (with different interference penalties) and action masking during training. The trade-off between safety and performance is evaluated

in thirty-times-longer-than-training episodes during testing, providing insights into the long-term effectiveness of the investigated methods.

## PROBLEM FORMULATION

An AEOS scheduling problem is considered, where a single satellite is tasked with imaging point targets (requests) on Earth's surface. The satellite is in a 600 km altitude with 45 degrees inclination orbit. Four actions (flight modes) are available to the agent at any given time: imaging, charging, downlinking, and reaction wheel desaturation. The mission goal is to maximize the number of imaged targets weighted by their priorities. The system is subject to two safety constraints: the battery level must remain above zero, and the reaction wheel angular velocities must not exceed their maximum limits. Then, the agent's objective is to find the sequence of actions that maximizes the cumulative reward.

### Partially observable semi-Markov decision process

The AEOS scheduling problem can be formulated as a Markov decision process (MDP), which is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, R, T, \gamma \rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $R$ is the reward function, $T$ the transition probability function, and $\gamma$ the discount factor. At each step, the agent must decide on an action $a \in \mathcal{A}$, given the current state $s \in \mathcal{S}$ of the environment. The environment then transitions to a new state $s' \in \mathcal{S}$ with probability $T(s'|s, a)$ and provides a reward $r = R(s, a)$. The goal is to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected cumulative discounted reward

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R(s_k, a_k) \mid a_k \sim \pi(s_k)\right] \forall s \in \mathcal{S} \tag{1}$$

where $k$ is the time step index.

The problem can be further expanded to a semi-MDP (sMDP) to account for variable action durations (such as imaging targets near and far away), which is shown to improve the policy performance.[21] In this case, the value function becomes

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^{\sum_{j=0}^{k} \Delta t_j} R(s_k, a_k) \mid a_k \sim \pi(s_k)\right] \forall s \in \mathcal{S} \tag{2}$$

where $\Delta t_j$ is the duration of action $a_j$.

Lastly, the problem can be extended to a partially observable sMDP (POsMDP) to account for the fact that the agent does not have access to the full state of the environment. Instead, is received observation $o \in \mathcal{O}$ from an observation space $\mathcal{O}$, which is related to the state through an observation function $Z : \mathcal{S} \times \mathcal{A} \to \mathcal{O}$. The policy is then defined as $\pi : \mathcal{O} \to \mathcal{A}$.

Therefore, the AEOS scheduling problem is modeled as a POsMDP, where:

- State space $\mathcal{S}$: Includes information about the satellite's states (such as position, battery charge fraction, etc), the environment states (such as target locations, ground station locations, etc), and internal flight software states.
- Action space $\mathcal{A}$: Contains $N + 3$ discrete actions, which are charging ($a_{\text{charge}}$), downlinking ($a_{\text{downlink}}$), reaction wheel desaturation ($a_{\text{desat}}$), and imaging one of the $N$ upcoming requests

($a_{\text{image},i}$ for $i = 1, \ldots, N$) in the unfulfilled list of requests $\mathcal{U}$. For this study, $N = 32$ with requests being ordered by their opportunity window.[21]

- Reward function $R$: Requests $\tau$ are defined by their priority ($\rho$) and location ($\mathbf{r}$), such that $\tau_i = (\rho_i, \mathbf{r}_i)$. The reward is proportional to the priority of successfully imaged targets:

$$R = \begin{cases} \rho_i, & \text{if } a = a_{\text{image},i} \text{ and target } \tau_i \in \mathcal{U} \text{ and } \tau_i \in \mathcal{F}' \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

  Rewards are only awarded for previously unfulfilled requests (re-imaging a target will not yield rewards). After imaging, the request is moved from the unfulfilled list $\mathcal{U}$ to the fulfilled list $\mathcal{F}$.

- Transition function $T$: The environment's dynamics are modeled through a high-fidelity simulator that accounts for the satellite's orbital dynamics, attitude dynamics, power system, reaction wheel dynamics, and ground station visibility.

- Observation space $\mathcal{O}$: Is a subset of the state space, which includes information about the satellite and the upcoming requests.[21]

- Observation function $Z$: The observation function is deterministic.

**Simulation environment**

The simulation environment was implemented in BSK-RL[22*], an open-source Python package that integrates Basilisk and Gymnasium, which is a widely used reinforcement learning environment interface. Basilisk[†] is an efficient high-fidelity spacecraft simulator with flight-proven software.[23]

When tasked with imaging a point target, the satellite must slew to point its camera towards the target location. The image was only successfully acquired if the relative angle and angular rate constraints were satisfied during the imaging task. The slew was controlled by a modified Rodrigues parameter steering law,[24] which was used to control the three reaction wheels. The reaction wheels were subject to maximum torque and angular momentum constraints. Successful imaging added data to the on-board data storage; when full, no more data could be added, and no more reward could be awarded.

The downlink action allowed the satellite to transmit the stored data to a ground station, freeing up space in the data storage. To succeed, the satellite must have access to a ground station and hold a nadir pointing attitude. Imaging, downlinking, and the use of reaction wheels consumed energy from the satellite's battery, in addition to a baseline power consumption. The charging action allowed the satellite to point its solar panels towards the sun to recharge the battery. Although the charging action could be performed at any time, it was only effective when the satellite was in sunlight. Also, the satellite passively charged when performing other actions, and its solar panels were exposed to sunlight.

When exposed to external torques, the reaction wheels' momentum can increase and reach their limit. The desaturation action allowed the satellite to desaturate the wheels by firing thrusters. The satellite reached a failure state if the battery level dropped to zero or if any of the reaction wheels

[*] https://avslab.github.io/bsk_rl/

[†] https://avslab.github.io/basilisk

exceeded their maximum angular momentum limits. An example script with the environment is available in the BSK-RL repository*.

## SHIELDED DEEP REINFORCEMENT LEARNING

### Training

The policy aforementioned can be obtained using reinforcement learning techniques, which improve the policy through interactions with the environment.[25] When the policy is represented by a neural network, the approach is referred to as deep reinforcement learning (DRL). The proximal policy optimization (PPO) algorithm[26] shows promise in solving the AEOS scheduling problem[21] and was used to obtain the policy. The PPO implementation from RLlib[27] (version 2.35.0) was used. The training was performed in the University of Colorado Research Computing Alpine high performance computing resource[28] using 32 AMD Epyc-7713 cores with 100 GB of RAM. Each policy was trained for up to 20M steps or 72 hours of wall-clock time.

Two fully connected layers of 2048 nodes each were used, with a learning rate of $3 \cdot 10^{-5}$, a training batch size of 3,000, and 10 epochs, along with a discount factor of 0.997. The gradient clipping was set to 0.5, PPO clipping was set to 0.2, and the generalized advantage estimation (GAE) was set to 0.95. Other parameters were set as the standard values in RLlib. The hyperparameters were used as they show good performance when compared to an optimal solver.[21]

Training episodes were limited to three orbits to allow for more frequent resets of the environment, resulting in a more randomized initial condition and a more diverse training set. The episodes were truncated after three orbits, so the value function was used to estimate the value of the next state.

### Shields

During training, the agent will be exposed to unsafe states, which can lead to terminal states and failures. Penalties can be added to the reward function every time the agent reaches the terminal state, discouraging unsafe behavior. However, this approach does not guarantee safety during training nor during testing. Alternatively, shields can be used to provide formal guarantees on safety. Two different shield formulations are considered in this work, one constructed using heuristics from expert knowledge of the problem,[11,29] and one constructed using formal methods.[30]

The handmade (*H*) shield is based on expert knowledge[20] and overrides the agent's selection to take action charge if it is not in eclipse and the battery charge is below a certain threshold. The battery threshold depends on a minimum energy threshold, $E_{\text{min}}$, and the time until the next eclipse. Therefore, the *H* shield will obtain the eclipse duration, $e_d$, and estimate the energy consumption during the eclipse, $E_e$, in that period with

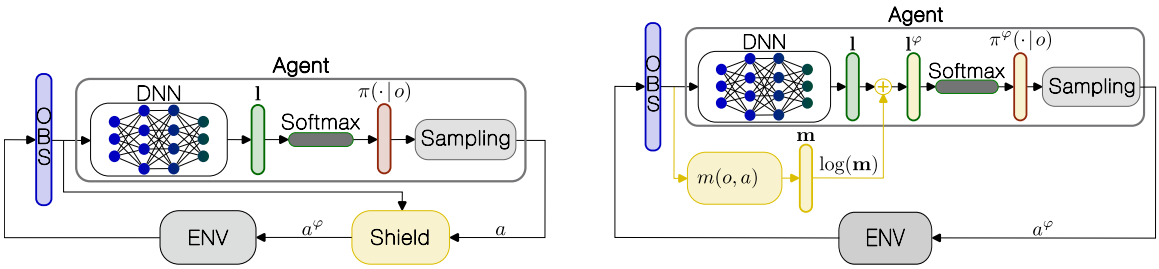
$$E_e = e_d P_d \tag{4}$$

where $P_d$ is an estimate of the satellite's power consumption. The minimum energy threshold, $E_t$, is calculated as

$$E_t = E_e + E_{\text{min}} - t_e P_c \tag{5}$$

where $P_c$ is the passive charging rate of the satellite, and $t_e$ is the time to the next eclipse. The *H* shield will also force the agent to enter the momentum management mode to desaturate the wheels

*https://avslab.github.io/bsk_rl/examples/training_with_shield.html




(a) Action replacement.

(b) Action masking.

**Figure 2**: Illustration of the differences between action replacement and action masking.

if any of the wheels' angular speed fraction is above a threshold $\hat{W}_t$. The $H$ shield will select the charge action if both conditions are met. Despite using expert knowledge, the $H$ shield might fail in edge cases that are not considered in the design. For this work, $E_{\text{min}} = 0.25$, $P_d = 1.0$, $P_c = 1.0$, and $\hat{W}_t = 0.7$.

The Q-Optimal ($Q$) shield is built using a safety-MDP[30] abstracted from the original POsMDP. The safety-MDP considers only states relevant to safety, such as battery levels and reaction wheel speeds. The probabilities of transitioning to unsafe states are obtained by sampling the high-fidelity simulator. The $Q$ shield is then constructed with dynamic programming. It obtains a policy $\pi_S^*$ that minimizes the probability of reaching an unsafe state. It allows only actions that have a probability $1 - p$ of leading to a safe state; if none is available, it uses $\pi_S^*$ to obtain the action more likely to lead to a safe state.

### Shielded training approaches

Shields can be used after training to provide safety guarantees during deployment. However, shields can also be used during training so that the policy learns while interacting with the shield. Two main approaches to incorporating shields during training are considered: action replacement and action masking. In action replacement, the shield overrides the agent's action with a safe action if the original action is deemed unsafe. In action masking, the shield is used to create a mask that prevents unsafe actions from being selected by the agent. Figure 2 illustrates the differences between action replacement and action masking. One of the main differences between the two approaches is that the selected safe action is used during the update step of the network in action masking, while the original action is used in action replacement.

When using action replacement, different interference penalties can be added to the reward function every time the shield overrides the agent's action. Four interference penalty levels are considered, which are $-0.01$, $-0.1$, $-1$, and $-10$. Also, three baseline policies are considered: unshielded training with no penalties, unshielded training with a failure penalty of $-10$ (which showed superior performance compare to other values of failure penalties in a similar scenario[21]), and a policy trained with an excessively large battery and no reaction wheel constraints (no safety constraints, focusing only on the imaging tasks). Table 1 summarizes the configurations of all trained policies.

## RESULTS

All trained policies were tested with and without shields in ninety-orbit-long episodes (equivalent to 6.3 days, being thirty times longer than training episodes) to evaluate the long-term trade-off

**Table 1**: Trained policies

| Case | Safety | Shield type | | | Shield mode | | Penalty | |
|---|---|---|---|---|---|---|---|---|
| | | Unshielded | $H$ | $Q$ | Replacement | Masking | Interference | Failure |
| $\pi_1$ | ✓ | ✓ | - | - | - | - | - | - |
| $\pi_2$ | ✓ | ✓ | - | - | - | - | - | -10 |
| $\pi_\infty$ | - | ✓ | - | - | - | - | - | - |
| $\pi_{H\text{-R}}$ | ✓ | - | ✓ | - | ✓ | - | - | - |
| $\pi_{H\text{-R}_{0.01}}$ | ✓ | - | ✓ | - | ✓ | - | -0.01 | - |
| $\pi_{H\text{-R}_{0.1}}$ | ✓ | - | ✓ | - | ✓ | - | -0.1 | - |
| $\pi_{H\text{-R}_1}$ | ✓ | - | ✓ | - | ✓ | - | -1 | - |
| $\pi_{H\text{-R}_{10}}$ | ✓ | - | ✓ | - | ✓ | - | -10 | - |
| $\pi_{Q\text{-R}}$ | ✓ | - | - | ✓ | ✓ | - | - | - |
| $\pi_{Q\text{-R}_{0.01}}$ | ✓ | - | - | ✓ | ✓ | - | -0.01 | - |
| $\pi_{Q\text{-R}_{0.1}}$ | ✓ | - | - | ✓ | ✓ | - | -0.1 | - |
| $\pi_{Q\text{-R}_1}$ | ✓ | - | - | ✓ | ✓ | - | -1 | - |
| $\pi_{Q\text{-R}_{10}}$ | ✓ | - | - | ✓ | ✓ | - | -10 | - |
| $\pi_{H\text{-M}}$ | ✓ | - | ✓ | - | - | ✓ | - | - |
| $\pi_{Q\text{-M}}$ | ✓ | - | - | ✓ | - | ✓ | - | - |

**Table 2**: Test case configurations

| Case | Unshielded | $H$ | $Q$ |
|---|---|---|---|
| $\pi_1, \pi_2, \pi_\infty$ | ✓ | ✓ | ✓ |
| $\pi_{H\text{-R}_\alpha},\ \alpha \in \{-, 0.01, 0.1, 1, 10\}$ | ✓ | ✓ | – |
| $\pi_{Q\text{-R}_\alpha},\ \alpha \in \{-, 0.01, 0.1, 1, 10\}$ | ✓ | – | ✓ |
| $\pi_{H\text{-M}}$ | ✓ | ✓ | – |
| $\pi_{Q\text{-M}}$ | ✓ | – | ✓ |

between safety and performance in each case. Table 2 summarizes the test cases.

Each case described in Table 1 was trained three times to account for randomness in training. Each resulting policy was tested with 20 random seeds for each configuration of request density (100, 500, 1000, 2000, 3000 requests), resulting in 100 test runs per policy (300 per training configuration). To avoid target depletion over time due to the extended episodes, a new random request was added to the unfulfilled list every time a target was successfully imaged. This approach also simulates a more realistic scenario where new imaging requests arrive dynamically during the satellite's operation.

### Unshielded testing

Figure 3 shows the aliveness during testing (after training) for policies trained without shields. Policy $\pi_2$ presented the highest survival rate, as expected, because it was trained with a failure penalty. $\pi_\infty$ shows the worst survival rate, as it was trained without safety constraints and did not

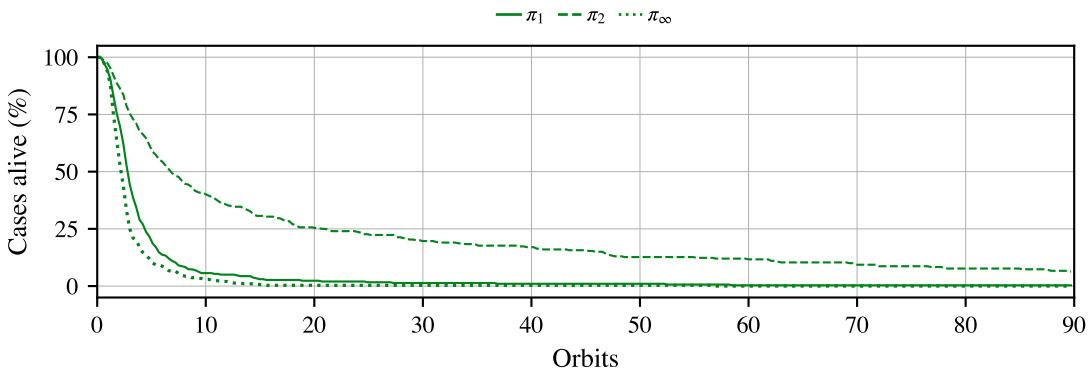



**Figure 3**: Aliveness during testing for policies trained and tested without shields.

learn how to manage its resources. $\pi_1$ falls in between the other two cases. Still, even the best policy presented less than 7% survival after ninety orbits, indicating the need for safety measures during deployment.

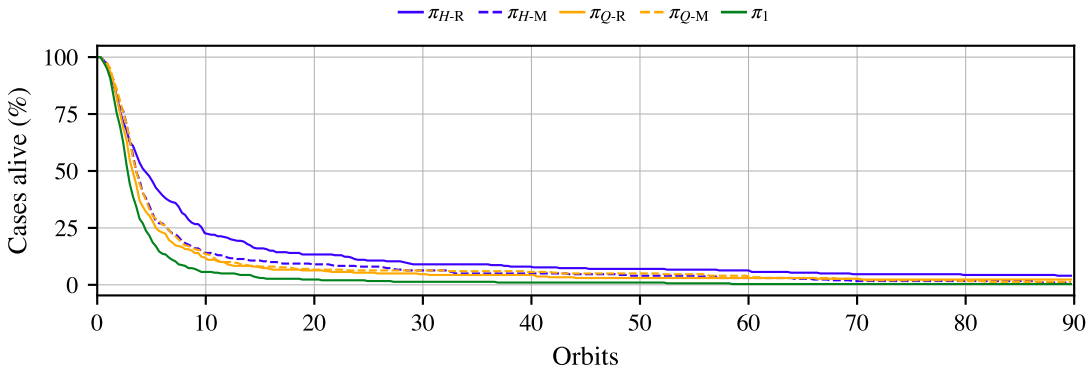
Figure 4a shows the aliveness during testing for policies trained with action replacement and action masking tested without shields ($\pi_1$ is also included for comparison). Although improvements are seen when compared to $\pi_1$, all cases show less than 5% survival after ninety orbits. On the other hand, Fig. 4b indicates that policies trained with action replacement with moderate interference penalties ($-0.1$ and $-1$) present significant improvements in survival rate when tested without shields. For example, cases $\pi_{H\text{-R}_1}$ and $\pi_{Q\text{-R}_1}$ showed 54.3% and 63.6% survival rate after ninety orbits, respectively. $\pi_{H\text{-R}_{10}}$ and $\pi_{Q\text{-R}_{10}}$, cases with high interference penalties, achieved even higher survivability, up to 75.6% and 78.3%, respectively.

Despite the safety improvements, these results indicate the need for shields during deployment. Nevertheless, these results provide insights into how different training methods can help the agent learn safety aspects of the problem, reducing safety violations when deployed without shields and shield interventions when deployed with shields.

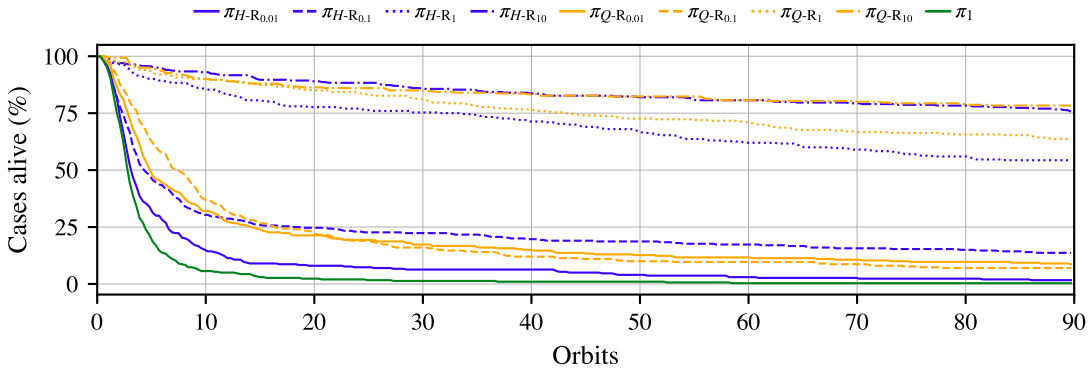
### Shielded testing

Figure 5 shows the performance of $\pi_\infty$ when tested with shields, together with shield interference. Shield interference is measured as the percentage of actions overridden by the shield during testing. The performance improves significantly when shielded with either shield ($\pi_\infty$ H and $\pi_\infty$ Q) compared to the unshielded case ($\pi_\infty$). The main drive for the performance improvement is the significant increase in survival rate, as satellites that reach a terminal state can no longer acquire rewards. Nevertheless, the shield interference was 15% and 25% on average for the $H$ and $Q$ shields, respectively, indicating a large reliance on the shields to maintain safety as expected. Although performant, this behavior indicates that the policy might be losing imaging opportunities due to the high interference from the shields (being forced to charge over regions with many requests, for example).

To better compare the impact of the different shields and shielding methods, Fig. 6 shows the relative performance of shielded policies $\pi_1$, $\pi_2$, and $\pi_\infty$. The relative performance is the average seed-by-seed ratio of the reward obtained by each policy compared to $\pi_2$ shielded with $Q$ ($\pi_2$ $Q$).




(a) Policies trained with action replacement and action masking tested without shields.




(b) Policies trained with action replacement with different interference penalties tested without shields.

**Figure 4**: Aliveness during testing for policies trained with shields and tested without shields.

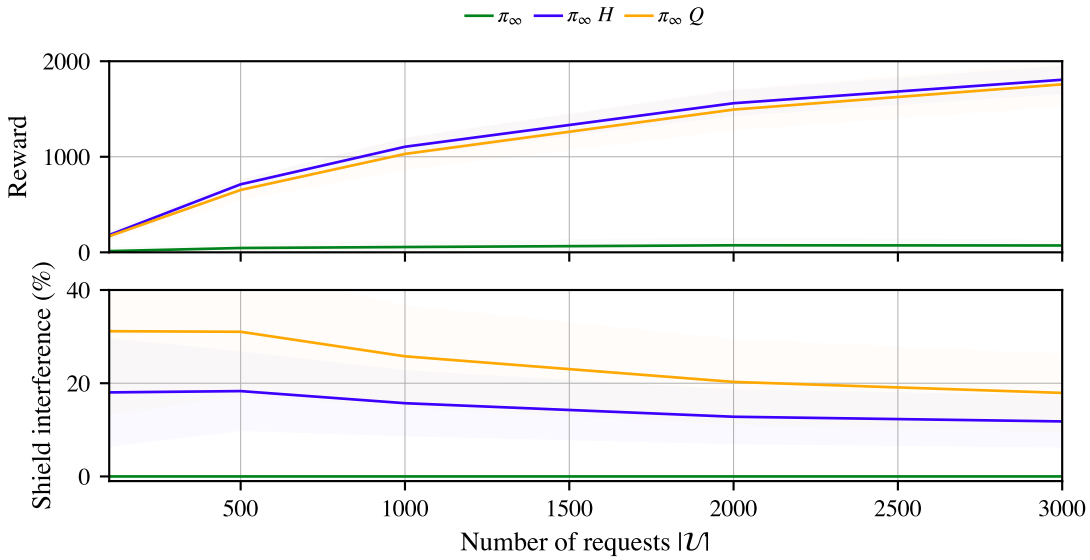



**Figure 5**: Performance of policies trained without shields when tested with shields.

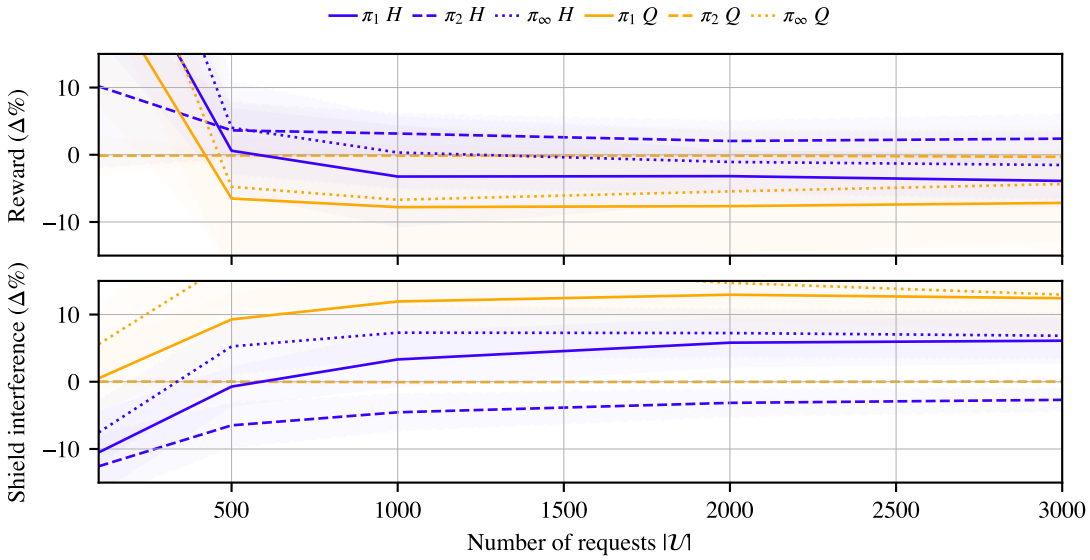



**Figure 6**: Relative performance comparison of shielded policies $\pi_1$, $\pi_2$, and $\pi_\infty$. $\pi_2$ $Q$ used as reference.

Overall, the same pattern is observed for both shields, where $\pi_1$ shows the worst performance, followed by $\pi_\infty$, and $\pi_2$ shows the best performance. In terms of shield interference, $\pi_2$ shows the lowest interference, and $\pi_\infty$ shows the highest interference. Interestingly, all cases seem to benefit from the $H$ shield compared to the $Q$ shield, showing higher rewards and lower interference. Despite being more conservative, the $Q$ shield provides formal safety guarantees.

Figure 7 presents the comparison between the cases trained with action replacement and action masking. Overall, the policies trained with masking showed superior performance but mixed results in terms of shield interference. This is expected as the masking approach uses the safe action selected by the shield during the update step of the network for backpropagation, allowing the agent to learn safer behaviors. Interestingly, $\pi_2$ $Q$ outperforms the cases trained with the $Q$ shield and shows comparable performance to those trained with the $H$ shield from 1,000 requests and above. Again, agents using the less conservative $H$ shield outperformed those using the $Q$ shield.

The results comparing the performance of policies trained with action replacement with different interference penalties are shown in Fig. 8. Overall, a moderate interference penalty of $-1$ yielded the best performance during testing, resulting in better results for both shields. Notably, cases with a high interference penalty of $-10$ showed a larger decrease in performance, with the case trained with the $Q$ shield failing to learn, indicating overly conservative behavior during training.

Overall, cases trained with interference penalties tended to outperform other methods. Adding a large failure penalty during unshielded training ($\pi_2$) also led to good performance, but such penalties can be sparse and provide a weak learning signal. In contrast, interference penalties provide more frequent feedback to the agent about unsafe behaviors.

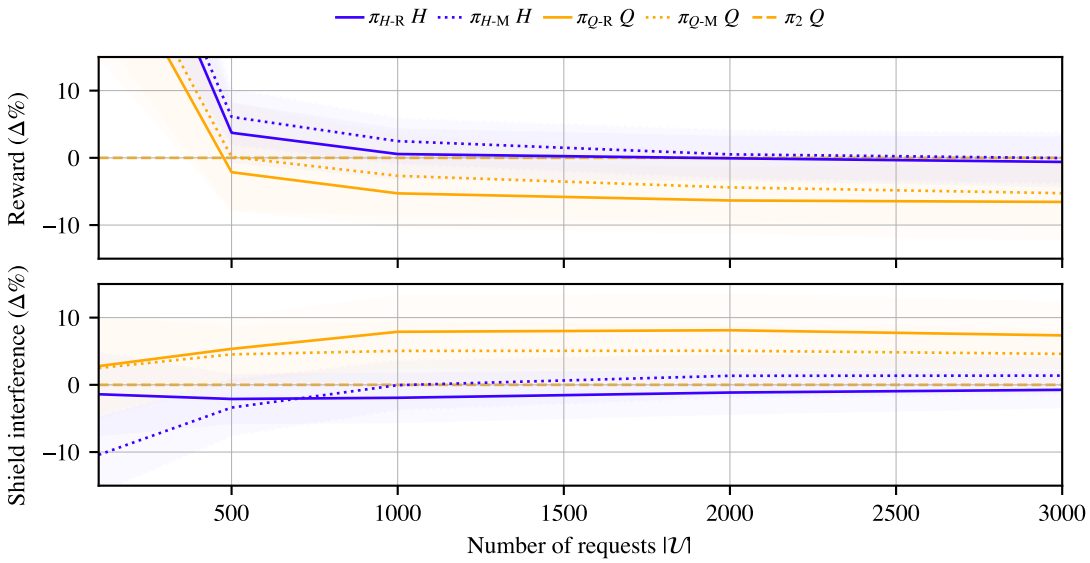



**Figure 7**: Relative performance comparison of action replacement and action masking training methods. $\pi_2$ $Q$ used as reference.

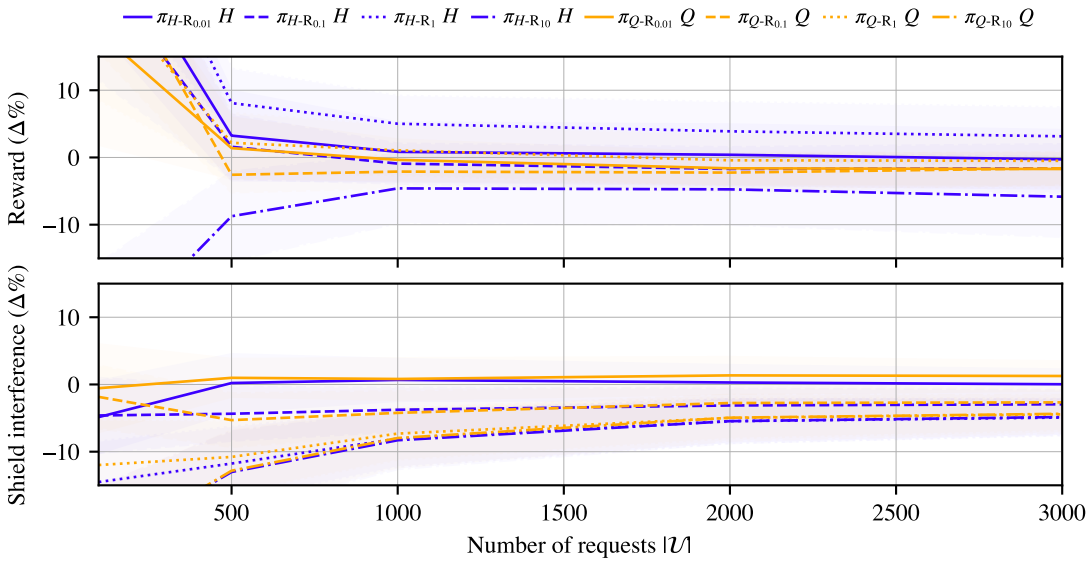



**Figure 8**: Relative performance comparison of action replacement training with different interference penalties. $\pi_2$ $Q$ used as reference.

### Elite cases

Although shields can provide safety, there is a possible trade-off with performance due to the interference with the agent's actions. Therefore, it is of interest to identify the performance decrease due to shielding. However, unshielded agents tend to reach terminal states quickly, bringing the average reward down. To provide a better comparison, only the cases that survived the entire ninety-orbit episode are considered, referred to as elite cases.[20] Figure 9 shows the relative performance of elite cases for policies $\pi_{H\text{-}\mathrm{R}_1}$ and $\pi_{Q\text{-}\mathrm{R}_1}$ when tested with and without shields.

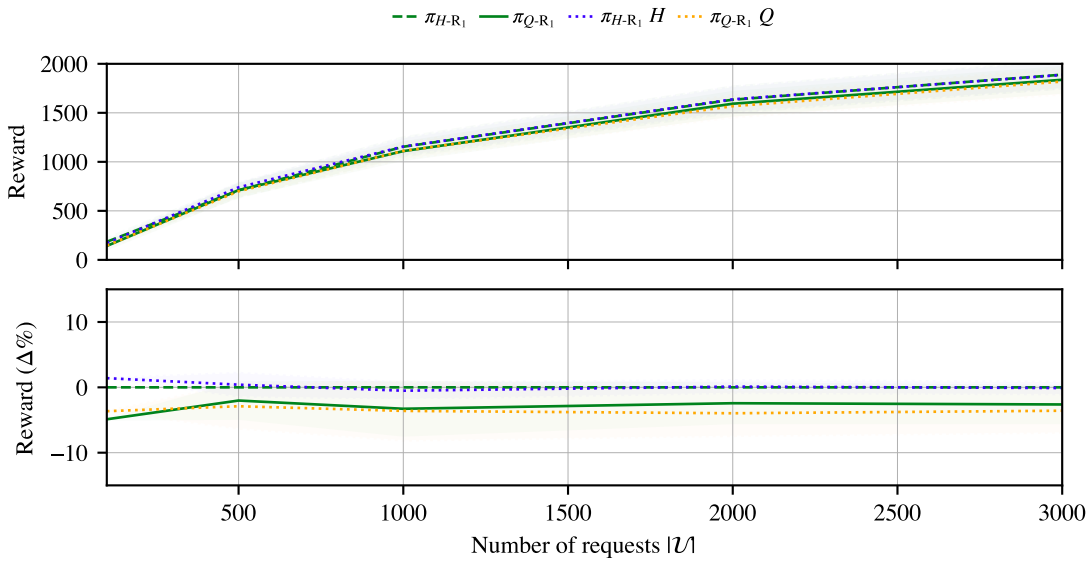



**Figure 9**: Performance of elite cases during testing. $\pi_{H\text{-}\mathrm{R}_1}$ used as reference.

The results indicate that all cases are similar in terms of rewards, with 0.26% and 0.25% difference on average for the shielded versions of $\pi_{H\text{-}\mathrm{R}_1}$ and $\pi_{Q\text{-}\mathrm{R}_1}$ compared to their unshielded versions, respectively. Despite the small performance decrease, all shielded cases achieved 100% survival, while unshielded cases achieved 54.3% and 63.6% survival for $\pi_{H\text{-}\mathrm{R}_1}$ and $\pi_{Q\text{-}\mathrm{R}_1}$, respectively. These results indicate that shields can provide safety guarantees with minimal performance decrease in the long run. This behavior is expected with the increase in episode duration, as the impact of failures becomes more significant; the agent that maximizes rewards is also the agent that minimizes failures.

## CONCLUSIONS

The use of different approaches to train deep neural networks with shields is investigated in the context of the autonomous agile Earth observation satellite (AEOS) scheduling problem. Two shield formulations are considered, a handmade shield based on expert knowledge and a shield constructed using formal methods. The two main shielded training approaches, action replacement and action masking, are explored.

The results indicate that all training methods lead to good-performing policies, except for action replacement with high interference penalties. Policies trained with moderate and high interference penalties during action replacement show the highest survivability when tested without shields,

incorporating safety aspects into the learned behavior. When tested with shields, these policies trained with moderate interference penalties present the best performance and the lowest shield interference. These results suggest the potential of using shields during training to enhance safety and performance during deployment.

Furthermore, the results indicate that shields can provide safety guarantees with minimal performance decrease in the long run. Elite cases that survived the entire testing episode showed an average performance decrease of less than 0.3% when shielded compared to their unshielded versions, while achieving 100% survival compared to 54.3% and 63.6% survival for the unshielded versions. These results highlight the potential of shields to enable safe and performant long-term operations of autonomous AEOS.

## ACKNOWLEDGMENTS

This work is partially supported by the Air Force Research Lab grant FA9453-22-2-0050. Also, this work utilized the Alpine high performance computing resource at the University of Colorado Boulder. Alpine is jointly funded by the University of Colorado Boulder, the University of Colorado Anschutz, Colorado State University, and the National Science Foundation (award 2201538). The authors also would like to acknowledge the support of Robert Reed and Dr. Morteza Lahijanian for their insights and help in developing and providing the $Q$ shield.

## REFERENCES

[1] M. Lemaître, G. Verfaillie, F. Jouhaud, J.-M. Lachiver, and N. Bataille, "Selecting and Scheduling Observations of Agile Satellites," *Aerospace Science and Technology*, Vol. 6, No. 5, 2002, pp. 367–381, 10.1016/S1270-9638(02)01173-2.

[2] N. Bianchessi, J.-F. Cordeau, J. Desrosiers, G. Laporte, and V. Raymond, "A heuristic for the multi-satellite, multi-orbit and multi-user management of Earth observation satellites," *European Journal of Operational Research*, Vol. 177, Mar. 2007, pp. 750–762, 10.1016/j.ejor.2005.12.026.

[3] P. Tangpattanakul, N. Jozefowiez, and P. Lopez, "A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite," *European Journal of Operational Research*, Vol. 245, Sept. 2015, pp. 542–554, 10.1016/j.ejor.2015.03.011.

[4] D. Eddy and M. J. Kochenderfer, "A Maximum Independent Set Method for Scheduling Earth-Observing Satellite Constellations," *Journal of Spacecraft and Rockets*, Vol. 58, Sept. 2021, pp. 1416–1429, 10.2514/1.A34931.

[5] X. Wang, G. Song, R. Leus, and C. Han, "Robust Earth Observation Satellite Scheduling With Uncertainty of Cloud Coverage," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 56, June 2020, pp. 2450–2461, 10.1109/TAES.2019.2947978.

[6] X. Wang, Y. Gu, G. Wu, and J. R. Woodward, "Robust scheduling for multiple agile Earth observation satellites under cloud coverage uncertainty," *Computers & Industrial Engineering*, Vol. 156, June 2021, p. 107292, 10.1016/j.cie.2021.107292.

[7] X. Zhao, Z. Wang, and G. Zheng, "Two-Phase Neural Combinatorial Optimization with Reinforcement Learning for Agile Satellite Scheduling," *Journal of Aerospace Information Systems*, Vol. 17, July 2020, pp. 346–357, 10.2514/1.I010754.

[8] A. Harris, T. Valade, T. Teil, and H. Schaub, "Generation of Spacecraft Operations Procedures Using Deep Reinforcement Learning," *Journal of Spacecraft and Rockets*, Vol. 59, March – April 2022, pp. 611–626, 10.2514/1.A35169.

[9] A. Herrmann and H. Schaub, "Reinforcement Learning for the Agile Earth-Observing Satellite Scheduling Problem," *IEEE Transactions on Aerospace and Electronic Systems*, 2023, pp. 1–13, 10.1109/TAES.2023.3251307.

[10] A. Herrmann, M. A. Stephenson, and H. Schaub, "Single-Agent Reinforcement Learning for Scalable Earth-Observing Satellite Constellation Operations," *Journal of Spacecraft and Rockets*, Nov. 2023, pp. 1–19, 10.2514/1.A35736.

[11] M. Stephenson and H. Schaub, "Reinforcement Learning for Earth-Observing Satellite Autonomy with Event-Based Task Intervals," *AAS Guidance and Control Conference*, Breckenridge, CO, Feb. 2–7 2024. Paper No. AAS 24-012.

[12] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe Reinforcement Learning via Shielding," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, Apr. 2018, 10.1609/aaai.v32i1.11797.

[13] I. ElSayed-Aly, S. Bharadwaj, C. Amato, R. Ehlers, U. Topcu, and L. Feng, "Safe Multi-Agent Reinforcement Learning via Shielding," *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, Richland, SC, International Foundation for Autonomous Agents and Multiagent Systems, 2021, p. 483–491.

[14] H. Krasowski, J. Thumm, M. Müller, L. Schäfer, X. Wang, and M. Althoff, "Provably Safe Reinforcement Learning: Conceptual Analysis, Survey, and Benchmarking," *Transactions on Machine Learning Research*, 2023.

[15] A. T. Harris and H. Schaub, *Spacecraft Command and Control with Safety Guarantees using Shielded Deep Reinforcement Learning*. Jan. 6–10 2020, 10.2514/6.2020-0386.

[16] A. Harris and H. Schaub, "Deep On-Board Scheduling For Autonomous Attitude Guidance Operations," *AAS Guidance, Navigation and Control Conference*, Breckenridge, CO, Jan. 30 – Feb. 5 2020. AAS 02-117.

[17] I. Nazmy, A. Harris, M. Lahijanian, and H. Schaub, "Shielded Deep Reinforcement Learning for Multi-Sensor Spacecraft Imaging," *2022 American Control Conference (ACC)*, 2022, pp. 1808–1813, 10.23919/ACC53348.2022.9867762.

[18] D. v. Wijk, K. Dunlap, M. Majji, and K. Hobbs, "Safe Spacecraft Inspection via Deep Reinforcement Learning and Discrete Control Barrier Functions," *Journal of Aerospace Information Systems*, Vol. 21, No. 12, 2024, pp. 996–1013, 10.2514/1.I011391.

[19] M. Stephenson, D. H. Prats, and H. Schaub, "Autonomous Satellite Inspection in Low Earth Orbit with Optimization-Based Safety Guarantees," *Proceedings of the 14th International Workshop on Planning and Scheduling for Space (IWPSS)* (C. Artigues, J. Jaubert, C. Pralet, and S. Chien, eds.), Toulouse, France, April 28–30 2025, pp. 209–218.

[20] L. Q. Mantovani and H. Schaub, "Performance Evaluation of Shielded Neural Networks for Autonomous Agile Earth Observing Satellites in Long Term Scenarios," *Proceedings of the 14th International Workshop on Planning and Scheduling for Space (IWPSS)* (C. Artigues, J. Jaubert, C. Pralet, and S. Chien, eds.), Toulouse, France, April 28–30 2025, pp. 112–121.

[21] M. A. Stephenson, L. Q. Mantovani, and H. Schaub, "Learning Policies for Autonomous Earth-Observing Satellite Scheduling over Semi-Markov Decision Processes," *Journal of Aerospace Information Systems*, May 2025, pp. 1–11, 10.2514/1.I011649.

[22] M. Stephenson and H. Schaub, "BSK-RL: Modular, High-Fidelity Reinforcement Learning Environments for Spacecraft Tasking," *International Astronautical Congress*, Milan, Italy, Oct. 14–18 2024.

[23] P. W. Kenneally, S. Piggott, and H. Schaub, "Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework," *Journal of Aerospace Information Systems*, Vol. 17, Sept. 2020, pp. 496–507, 10.2514/1.I010762.

[24] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*. Reston, VA: AIAA Education Series, 4th ed., 2018, 10.2514/4.105210.

[25] R. S. Sutton and A. Barto, *Reinforcement learning: an introduction*. Adaptive computation and machine learning, Cambridge, Massachusetts London, England: The MIT Press, second edition ed., 2020.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," Aug. 2017.

[27] E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica, "RLlib: Abstractions for Distributed Reinforcement Learning," June 2018, 10.48550/arXiv.1712.09381.

[28] University of Colorado Boulder Research Computing, "Alpine," 2023. University of Colorado Boulder, 10.25811/K3W6-PK81.

[29] A. Herrmann and H. Schaub, "A comparative analysis of reinforcement learning algorithms for earth-observing satellite scheduling," *Frontiers in Space Technologies*, Vol. 4, Nov. 2023, p. 1263489, 10.3389/frspt.2023.1263489.

[30] R. Reed, H. Schaub, and M. Lahijanian, "Shielded Deep Reinforcement Learning for Complex Spacecraft Tasking," *2024 American Control Conference (ACC)*, IEEE, July 2024, p. 2331–2337, 10.23919/acc60939.2024.10644855.