AAS 24-208

REINFORCEMENT LEARNING FOR SATELLITE AUTONOMY UNDER DIFFERENT CLOUD COVERAGE PROBABILITY OBSERVATIONS

Lorenzzo Quevedo Mantovani[*], Yumeka Nagano[*], Hanspeter Schaub[†]

This paper investigates the use of Deep Reinforcement Learning (DRL) to address the Agile Earth Observing Satellite (AEOS) scheduling problem when considering cloud coverage. Earth Observing satellites play a crucial role in acquiring data from Earth using sensing instruments. However, clouds are a constant presence on Earth and impose a major challenge by obscuring targets and prohibiting data collection. In this study, the AEOS scheduling problem is modeled as a Partially Observable Markov Decision Process (POMDP), and a DRL-based approach is proposed to obtain a policy for on-board decision-making considering cloud coverage uncertainty. Five different observation capabilities of cloud coverage are considered, and two distinct reward models – binary and linear – are tested based on the percentage of cloud coverage. Additionally, two distinct cloud models are used for training, one based on historical cloud data and another based on a stochastic cloud model. A high-fidelity simulation environment is used to train and test the agents' performance under realistic conditions. The results demonstrate that agents with the ability to observe cloud information outperform those without such capabilities. Furthermore, agents trained in an environment with clouds but incapable of observing cloud information show comparable performance and better resource management than agents trained in a cloud-free environment. Besides, agents trained with the stochastic cloud cover model show comparable or superior performance than those trained with historical cloud data.

INTRODUCTION

This paper investigates using Deep Reinforcement Learning to solve the Agile Earth Observing Satellite (AEOS) scheduling problem when considering cloud coverage. Earth observing satellites are used to acquire data from Earth using sensing instruments, while agile indicates satellites that can maneuver along- and across-track, increasing their flexibility to collect data. In this context, the scheduling problem consists of determining the sequence of tasks to be executed by the satellite to maximize the mission objective while considering the satellite's constraints. Traditionally, the ground segment performs the planning and scheduling steps using operators or optimization algorithms. However, these approaches have limitations, such as the need to re-plan when new requests are added, the inability to adapt to changing conditions quickly, and the difficulty to account for uncertainties. Such uncertainties can be weather conditions affecting image quality, such as snow covering ground areas of interest, humidity affecting the quality of the image, or the presence of clouds obscuring the target and prohibiting the satellite from collecting data. For example, the

[*] PhD Student, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado Boulder, Boulder, CO, 80303.

[†] Professor and Department Chair, Schaden Leadership Chair, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado Boulder, 431, Colorado Center for Astrodynamics Research, Boulder, CO, 80309.

annual mean cloud cover in LandSat ETM+ acquisitions was more than 30%, even when using a Long-term Acquisition Plan (LTAP), which also tries to reduce the amount of clouds in the acquired images.[1]

Traditional optimization methods have been shown to solve the scheduling problem for non-agile and agile Earth Observing Satellites (EOS). Lemaître et al. (2002) investigated the AEOS and showed that even with simplification assumptions, the problem is NP-hard;[2] the authors also compared a greedy algorithm, dynamic programming, constraint programming, and local search algorithm to solve the problem. Over time, other methods have been proposed to solve the scheduling problem for agile satellites. Bianchessi et al. (2007) investigate the use of heuristics for multi-satellite systems while Tangpattanakul et al. (2015) employ local search algorithm to maximize the total reward obtained by an agile satellite while minimizing the difference in reward of individual users that share the platform.[3,4] Eddy and Kochenderfer (2021) propose the use of an infeasibility-based graph, where the best solution corresponds to the maximum independent set of the graph.[5] The method is investigated with up to 10,000 requests and 24 satellites; the satellite is assumed to slew at a constant rate, and power constraints are not considered. Wang et al. (2021) provide an overview of the scheduling problem for the AEOS and the different methods proposed to solve it.[6]

Still, these optimization methods are brittle to initial conditions, present challenges to incorporate more complex constraints and require total or partial replanning when new requests are added to the system. The use of machine learning techniques has been proposed to mitigate these problems. More specifically, Deep Reinforcement Learning (DRL) has shown the ability to effectively solve the scheduling problem under complex constraints while presenting high adaptability. A DRL approach has been investigated to solve the multi-AEOS scheduling problem,[7] and different information-sharing methods have been analyzed to create collaboration among agents.[8] DRL has also been applied to the planning and scheduling of spacecraft for small-body science operations.[9] In-depth comparisons of different DRL algorithms for the scheduling of Earth-observing satellites were also performed by Herrmann and Schaub (2023) while Stephenson and Schaub (2024) provide a comparison between DRL and Mixed Integer Programming (MIP) under different target distributions.[10,11] Yet, these works do not consider cloud coverage in the AEOS scheduling problem.

Different methods were investigated to mitigate the problem imposed by clouds and incorporate it into the scheduling process. Wang te al (2016), for example, account for clouds in a single non-agile satellite scheduling problem;[12] the authors introduce clouds as a stochastic process and use a chance constraint programming (CCP) method combined with a sample approximation to transform the problem into an integer linear programming and a branch and cut algorithm used to solve it. In the work of Wang et al. (2019), clouds are also modeled as stochastic processes, but the authors introduce the possibility of targets being imaged more than once to increase the probability of being successfully imaged due to cloud coverage.[13] Additionally, Wang et al. (2020) studies the scheduling of multiple satellites considering cloud coverage.[14] Their proposed approach using the branch-and-price algorithm is compared with the CPLEX and demonstrated to solve the problem faster. Still, satellites are considered non-agile, and only a small number of 400 requests are considered.

In the work of Valicka et al. (2019), the authors introduce a MIP approach to solve the scheduling problem from one to an arbitrary number of satellites with weather uncertainty.[15] More specifically, they introduce cloud coverage as a function of time and demonstrate that their stochastic approach outperforms deterministic models. The quality of the acquired image is assumed to be inversely proportional to the cloud coverage of the target. Additionally, the authors demonstrate the feasibility of solving their problem formulation using off-the-shelf solvers such as the CPLEX solver for MIP. With a different approach, Hadj-Salah et al. (2019) propose using DRL to minimize the time required to image a given region of interest subject to cloud coverage.[16] The authors model the problem as a Partially Observable Markov Decision Process (POMDP) and use an Advantage Actor Critic (A2C) algorithm to find the policy. Predicted cloud coverage is passed to the agent to decide whether to image the targets; Cloud data is extracted from the ERA-Interim dataset. The authors' findings indicate that the policy obtained with A2C outperforms a heuristics-based approach. Nevertheless, their work does not consider satellite dynamics or power and storage constraints.

An approach based on budgeted uncertainty is proposed by Wang et al. (2020) to deal with cloud coverage uncertainty.[17] Additionally, the authors propose a heuristic that outperforms standard solvers. Their work considers instances with up to 300 targets and is limited to non-agile satellites. Wang et al. (2021) expand the use of budget uncertainty with AEOS and show the possibility of solving it using column generation and simulated annealing (SA) methods.[18] The combination of column generation with SA and CPLEX takes an average of 770 and 1900 seconds, respectively, for 250 targets. Still, the number of targets in the problem is limited to 300. The authors comment that limited attention was given to the AEOS scheduling problem with cloud uncertainty and highlight limitations in previous work due to computation requirements in the context of cloud uncertainty and consideration of only binary reward models, where rewards are calculated only based on a cloud coverage threshold and not proportional to it. In future work, the authors propose investigating real-time scheduling. Zhang et al. (2022) highlight that previous research do not account for real cloud information and cloud change over time and proposes an on-board cloud detection and re-plan.[19] The satellite is assumed to have a cloud-detecting sensor in addition to the optical camera used to image targets; the satellite can decide to continue with the original plan, abandon the image opportunity, or re-plan based on new window opportunity depending on the actual cloud coverage.

In addition to the simplification of cloud models and associate rewards, Gu et al. (2022) also mention the lack of use of real cloud information as a limitation seen in previous studies.[20] Therefore, the authors introduce a method to allow re-planning based on reliable cloud forecasts provided by a neural network leveraging previous imaging taken by the satellite. The initial and proactive plan is formulated using CCP (similar to the approach used by Wang et al. (2016)).[12] Then, the full plan horizon is divided into smaller intervals for re-plan based on the updated cloud forecast. A linear reward model is introduced based on actual cloud coverage, where the reward assigned to a request is inversely proportional to the cloud coverage. The author investigates replanning horizons of 1, 2, and 3 hours and up to 200 observation targets; their results indicate that the proposed replanning method outperforms other techniques.

A CCP approach is also used by Han et al. (2023) to investigate the multi-agile EOS problem under cloud uncertainty.[21] CCP is used with a sample approximation and an improved SA algorithm. The cloud coverage is simplified to be a binary case of cloud or no cloud and is represented by a stochastic process. The research results indicate that the proposed method can obtain higher rewards while taking less time to find a solution (ranging from a few seconds to a few minutes) when compared to a genetic algorithm, adaptive large neighborhood search, and bidirectional dynamic programming-based iterated local search. Still, the maximum number of targets considered is 950, and the maneuver times between different targets are simplified.

Candela et al. (2023) consider a satellite with a lookahead instrument to collect information about upcoming targets and the presence of clouds.[22] Then, the algorithm decides where to target the radar (the primary instrument) to avoid or target clouds, depending on the clouds' characteristics and mission goals. Their results indicate that having information about upcoming targets can significantly improve the system's performance. Still, their work does not consider the full satellite dynamics to account for maneuver time between targets and more realistic power consumption.

A DRL approach is used to tackle the multi-AEOS problem with targets being obscured by clouds by Naik et al. (2024).[23] The authors employ the Basilisk* software (see Kenneally et al. (2020)[24]) to have a high-fidelity simulation environment and consider targets with stochastic cloud coverage. The problem is modeled such that the satellite can image one of the targets, downlink the data to a ground station, or enter a battery charge mode. Their result indicates the capability of the agent to learn how to react to cloud coverage on the fly.

Overall, these works do not consider high-fidelity satellite dynamics to account for realistic maneuver time between targets and constraints such as power consumption. Also, previous methods tend to be limited in their ability to quickly re-plan based on cloud information and other states of the spacecraft, such as battery levels. Additionally, few works consider the use of historical and realistic cloud data to evaluate the system.

Therefore, this paper presents the use of DRL to obtain a policy for on-board decision-making to solve the AEOS scheduling problem considering cloud coverage. Cloud data from the ERA5 dataset is used for training and testing the agent.[25] A high-fidelity simulation environment is used to test the agent's performance under realistic conditions such as maneuver time and power consumption. The problem formulation allows to incorporate cloud information provided by ground stations, relays of other satellites, on-board weather forecast tools (such as the one investigated by Gu et al. (2022)), or cloud-sensing instruments (as proposed by Zhang et al. (2022) and Candela et al. (2023)).[19,20,22] Further, two distinct reward models are considered based on the percent of cloud coverage, binary and linear. The number of requests considered in this work is also higher than in most previous works, varying from 1,000 to 10,000, allowing for a more comprehensive analysis of the system's performance.

Therefore, the main contributions of this work are the use of DRL to solve the AEOS scheduling problem considering cloud coverage, the analysis of the impact of different observation capabilities of cloud coverage in the agent's performance, and the investigation of two different reward models based on cloud coverage percentage. Hence, the paper is organized as follows: we formulate the problem as a POMDP in Section Problem Formulation. Next, we present the simulation and training environments in the Solution Approach Section. In Results, we show results and discuss findings. Finally, we conclude the work in the Conclusion.

PROBLEM FORMULATION

The satellite scheduling problem can be modeled as a POMDP to be solved by Reinforcement Learning. Therefore, first, we present the problem as a POMDP. Then, we describe the two considered reward functions.

Partially Observable Markov Decision Process

A Markov Decision Process (MDP) can represent a sequential decision process under the Markov assumption, where the next state s' depends only on the current state s and action a taken.[26] In the case where the agent cannot observe all states of the environment, the problem is modeled as a

*https://hanspeterschaub.info/basilisk

Table 1. Observation Space of the Satellite.

Parameter	Normalization	Description
${}^{\mathcal{P}}\hat{r}_{\mathcal{B}/\mathcal{N}}$	6378136.6	Position of the satellite with respect to the inertial frame $\mathcal{N}$, expressed in the planet frame $\mathcal{P}$ and normalized by the Earth's radius.
${}^{\mathcal{P}}\hat{v}_{\mathcal{B}/\mathcal{P}}$	7616.5	Velocity of the satellite with respect to the planet frame, expressed in the planet frame and normalized by $\lvert {}^{\mathcal{P}}\mathbf{v}_{B/N} \rvert$.
$\hat{c}_{\mathcal{B}/\mathcal{P}}$	π	Orientation of the satellite's sensing instrument with respect to the planet frame.
${}^{\mathcal{P}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{P}}$	0.03	Angular velocity of the satellite with respect to the planet frame, expressed in the planet frame.
P_b	$P_{b_{max}}$	Battery charge level.
Φ_s	π	Angle between solar panels and sun vector.
e_{start}	T	Next eclipse start time, normalized by the orbital period T
e_{end}	T	Next eclipse end time
ρ_i	-	Priority of target i.
c_{f_i}	-	Cloud coverage forecast of target i.
σ_i	-	Standard deviation of the cloud coverage forecast of target i.
t	t_{episode}	Time normalized by the episode duration t_{episode}.

POMDP. For the satellite scheduling problem, the agent observes only states of the environment that are considered to be relevant to the decision-making process. Therefore, the problem is formulated as a POMDP defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \mathcal{O}, Z, \gamma \rangle$, where:

- State Space $\mathcal{S}$: The states available in the simulation, including the satellite's position, velocity, angular position, angular velocity, targets' information, and cloud coverage;
- Action Space $\mathcal{A}$: The action space has length 33, which includes actions charge and 32 imaging actions, corresponding to the upcoming 32 target window opportunities;
- Transition Function $T(s'|s,a)$: Transition probabilities are generated by a deterministic model with the satellite's dynamics from a high-fidelity simulator;
- Reward function R: Maps states and actions into rewards. Two distinct reward functions are tested;
- Observation Space $\mathcal{O}$: Subset of the state space which the agent can observe. The observation space includes states that are relevant for decision-making in the problem;
- Observation function Z: The satellite can observe without uncertainty the states in the observation space;
- γ: Discount factor.

The satellite observation space is described in Table 1, showing the states the satellite can observe and the normalization used to keep the values between -1 and 1 (a good practice when using neural networks).

There are 2 terminal states that are reached either when the episode ends or when the battery charge level reaches zero ($P_b \leq 0$). As mentioned, the transition probability function is deterministic, such that $T(G(s,a)|s,a) = 1$, where G is the generative model and $s' = G(s,a)$.

A high-fidelity simulation environment is used as the generative model ($s' = G(s,a)$) to train and test the agent under realistic scenarios, considering maneuver times and power consumption. Therefore, the simulation is implemented using BSK-RL*,[27] which is an open-source Python package focused on spacecraft tasking. BSK-RL combines the Gymnasium environment with Basilisk, which is a high-fidelity astrodynamics simulator with flight-proven software. Basilisk allows the simulation of spacecraft dynamics, sensors, actuators, and environment models.[24]

Reward Functions

Two distinct reward functions are tested, inspired by previous research accounting for cloud coverage. First, a binary reward model is considered, where the target is deemed either occluded by clouds or not. The reward is given by

$$R = \begin{cases} \rho_i & \text{if } c_{p_i} \leq c_t \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where c_{p_i} is the true cloud coverage of target i, c_t is a threshold value, and ρ_i is target i priority. Sometimes, the binary reward model is referred to as a simplification of the problem since partial credit could be assigned to targets partially occluded by clouds.[20] Therefore, a second reward model is considered, where the reward is proportional to the cloud coverage of the target until a given threshold, given by:

$$R = \begin{cases} \rho_i(1 - \frac{c_{p_i}}{c_t}) & \text{if } c_{p_i} \geq 0 \text{ and } c_{p_i} \leq c_t \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Although these two reward models are considered in this paper, other reward functions can be easily employed in the proposed DRL framework to better meet the needs of satellite operators and other stakeholders. Figure 1 provides a visual representation of the two reward models. In both cases, rewards are only awarded to targets not imaged before; imaging a target more than once won't provide additional rewards.

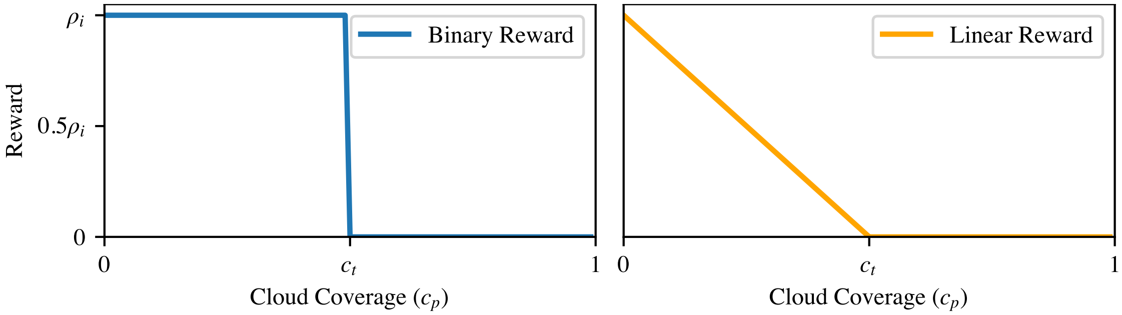

Figure 1. Binary and Linear Reward Models as a function of c_p for a given c_t.

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

SOLUTION APPROACH

Simulation and Training Environment

The simulation model implemented in Basilisk for this paper accounts for a baseline power consumption at all times and an additional power consumption to image targets (instrument consumption). Additionally, maneuvering towards a target or sun to charge the battery requires using reaction wheels to provide torques, impacting power consumption. The satellite will passively charge during the simulation whenever the solar panels are facing the sun – and not in eclipse –, even when performing another action. When taking charge actions, the spacecraft maneuvers to align the solar panels' faces with the sun to maximize energy generation. Additionally, when the satellite is tasked with an imaging action, it is not guaranteed to succeed since the simulator will check if the imaging angle and the relative angular rate are within the specified limits. An example script named Cloud Environment is available at the BSK-RL GitHub repository[*], which includes the simulation model, satellite dynamics, target models, and linear reward function used in this work.

The simulation includes power as a constraint, so the battery level should not reach zero. Therefore, the charge action is added as an option to the satellite. Although not considered here to focus on the cloud coverage aspect of the problem, the simulation model can be expanded to include other constraints such as memory and reaction wheels speed, and actions such as data downlink and reaction wheels momentum management.[8,28] Also, the proposed framework focuses on one satellite, but the approach can be extended to accommodate multiple satellites using the intent-sharing communication method to create collaboration among agents, as discussed by Stephenson and Schaub (2024).[28]

The solution approach is based on using DRL to obtain a policy to maximize cumulative rewards. Then, the RLlib Ray Python package is used to obtain the policies. RLlib is a scalable reinforcement learning library that provides a unified API for a variety of reinforcement learning algorithms. The policy is trained using the Asynchronous Proximal Policy Optimization (APPO) algorithm, which is a variant of the Proximal Policy Optimization (PPO).[29] The training algorithm was deployed in the University of Colorado Research Computing (CURC) using 32 cores.[30]

Five agents with different observation capabilities of cloud coverage were considered:

- CloudSat c_f and σ: The agent can observe the cloud coverage forecast (c_f) and standard deviation of the cloud coverage forecast (σ).
- CloudSat c_f: The agent can observe the cloud coverage forecast (c_f).
- CloudSat σ: The agent can observe the standard deviation of the cloud coverage forecast (σ).
- CloudSat: The agent cannot observe cloud coverage information but is trained in an environment with clouds (reward function still considers the true cloud coverage).
- NormalSat: The agent cannot observe cloud coverage information and is trained in an environment without clouds (all targets have zero cloud coverage).

These five agents were trained with the two different reward models and two cloud models, resulting in 20 distinct training runs and policies.

[*] https://avslab.github.io/bsk_rl/examples/cloud_environment.html

While the agent should learn to charge when required during training and prepare for eclipse periods, the obtained policy can still lead to unsafe states and failures. For a long-term deployment, the policy would be combined with shields to guarantee safety.[31–34] However, shields can interfere with the selected actions and the agents' performance. Therefore, shielded neural networks are not used during training or deployment to focus on the agent's relative performance with different observations and scenarios.

Targets and Cloud Model

The targets are uniformly distributed around the world, while the number of requests (number of targets to be imaged) changed randomly during training, varying from 1,000 up to 10,000, which impacts the target density. Although a point-and-shoot model is used, regions of interest can be decomposed into points.[5] Additionally, each target has an individual cloud coverage associated with it.

The true cloud coverage of the targets was modeled using two different scenarios for training: one where the cloud coverage is randomly assigned to the targets and another where the cloud coverage is based on historical data. The historical cloud coverage data was obtained from the ERA5 dataset from the total cloud coverage parameter, which contains hourly data on cloud coverage probability since 1940.[25] The dataset combines different measurement sources using an assimilation process to have a global coverage with discretization of 15 minutes of latitude and longitude. The targets' latitude and longitude were used to interpolate the cloud coverage from the dataset. Figure 2 shows the cloud coverage extracted from the ERA5 dataset for the globe and the result of the interpolation for the targets.

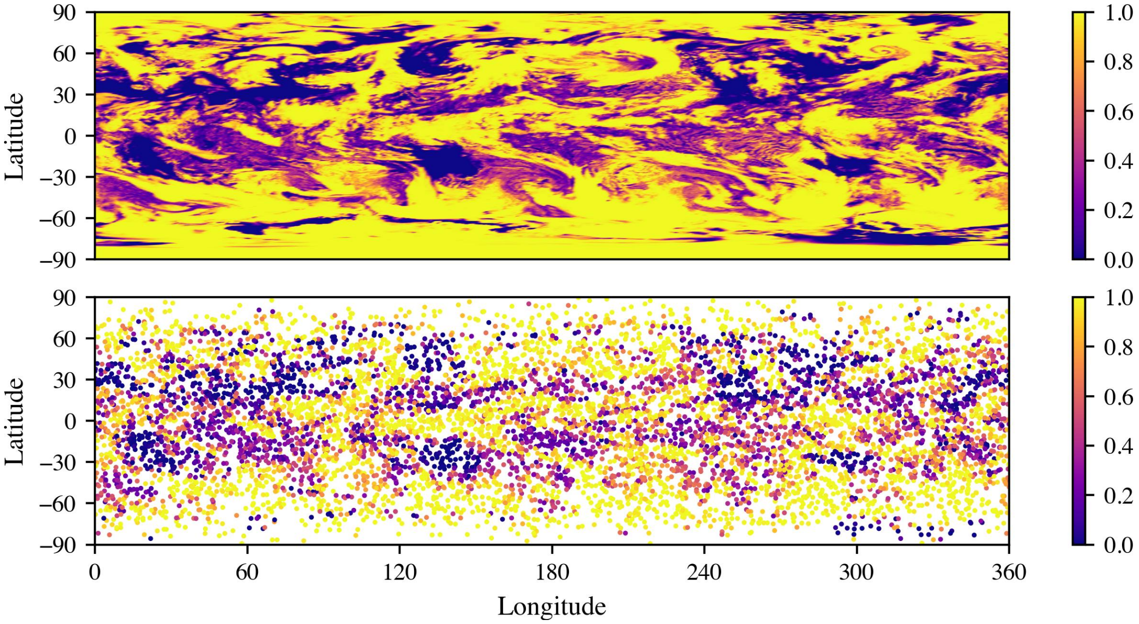

Figure 2. Comparison between cloud coverage from the ERA5 dataset from the year 2023 (top) and targets' cloud coverage obtained interpolating the data (bottom).

A stochastic cloud model is desirable for training when compared to historical cloud data from datasets since the latter requires more storage space and is limited in time. Therefore, the stochastic cloud coverage scenario is approximated with a uniform distribution, with a mean similar to the

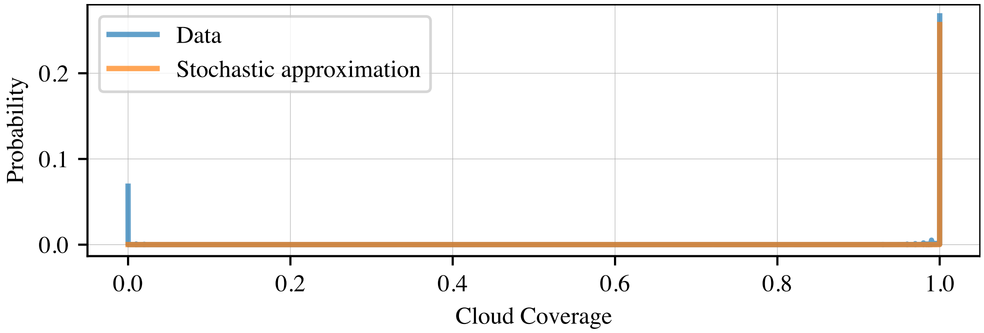

Figure 3. Cloud coverage probability from ERA-5 data (blue) and using the stochastic cloud coverage from Equation (3) **(orange).**

historical data. In this case, the cloud coverage probability is randomly assigned to the targets following

$$c_p = \max\left(0, \min\left(\mathcal{U}(0, 2\mu), 1\right)\right) \tag{3}$$

such that the output of cloud coverage is clipped between 0 and 1. This stochastic model preserves the mean cloud coverage of the data while generating more targets completely occluded by clouds than free of clouds, similar to the distribution seen in the historical data. The approximation given by Equation (3) aims at providing a simple and coarse representation of the cloud coverage scenario; more complex models can be used instead if needed. A mean of $\mu = 0.674$ was obtained considering three years of hourly data from the ERA5 dataset. Figure 3 compares the cloud coverage probability from the ERA5 dataset and the stochastic cloud coverage scenario. A main difference from the stochastic cloud coverage scenario is the lack of spatial correlation between the cloud coverage probabilities of the targets.

Both stochastic cloud coverage and historical data were used to train the agents and are compared in the results section. For training, hourly cloud data was randomly sampled from the years 2011 to 2022. All agents were tested with historical cloud data sampled from the ERA5 dataset from 2023. In all scenarios, the cloud coverage was assumed to be constant during the training and testing episodes (not time-varying).

The cloud coverage provided by c_p represents the true cloud coverage of the target and is used to calculate the reward, as seen in Equations (1) and (2); this information is never provided to the agent. Instead, the agent uses the cloud coverage forecast, c_f, when observed, to decide whether to image the target or not. The cloud coverage forecast is obtained from the cloud coverage, c_p, and the standard deviation of the cloud coverage forecast, σ, using a Gaussian distribution given by

$$c_f = \max(0, \min(\mathcal{N}(c_p, \sigma), 1)) \tag{4}$$

where σ is also given by a uniform distribution $\mathcal{U}(0.01, 0.05)$ for each target. In this context, σ represents confidence in the cloud coverage forecast. The range of σ was chosen to represent a reasonable range of confidence in the forecast but can be adjusted to represent different scenarios. As mentioned before, the proposed framework can be easily adapted to use cloud coverage forecasts from other sources, such as neural networks or weather forecast tools.

Table 2. Satellite Parameters

Altitude	500 km
Mass	330 kg
Inertia	$[121, 98, 82]$ kg m^2
Battery capacity	160 W
Initial battery charge fraction	$[0.4, 0.6]$
Instrument power draw	30 W
Relative angle limit for imaging	28°
Relative angular rate limit for imaging	0.01 rad/s
c_t for binary case	0.2
c_t for linear case	0.7
Number of requests	$[1000, 10000]$
σ_i	$[0.01, 0.05]$
ρ_i	[0, 1]

Table 3. Training Parameters

Number of workers	32
Number of raining steps	$5 \cdot 10^6$
Learning rate	$3 \cdot 10^{-5}$
Training bath size	$10,000$
Minibatch size	250
Number of SGD iterations	50
Neural Network	2 layers with 512 neurons each
Discount factor	1.0
Failure penalty	0

Satellite and training parameters

The satellite parameters used for the simulation are shown in Table 2 in addition to the initialization and targets' parameters. Parameters not listed here, such as control gains, are set to the standard values available in BSK-RL.

The training parameters for APPO were based on the work of Herrmann and Schaub (2023),[10] which provides an in-depth analysis of hyperparameters selection for the imaging satellite case. The parameters are shown in Table 3. All other parameters were set as the standard values in RLLib. The training was terminated when a policy reached $5 \cdot 10^6$ training steps or 24 hours of wall clock time. Further, when the satellite battery charge level reaches zero, the episode is terminated, but no penalty is assigned to it.

After training, evaluating the neural network at each decision-making step takes about 10 milliseconds in a MacBook with an M2 Pro chip and 16Gb of RAM, showing its potential for on-board real-time decision-making.

RESULTS

All 20 trained policies were tested in a similar environment. The test environments varied the number of requests from 1,000 up to 10,000 (in increments of 1,000), with the true cloud coverage being assigned based on randomly hourly data sampled from the year 2023 of the ERA5 dataset. Each agent was tested on 100 three-orbit long runs for each number of targets, and the average results are presented. The performance of each agent is analyzed based on the average reward obtained over the 100 runs, the number of acquired images, and the cloud-free metric, which relates the number of cloud-free images to the total number of images acquired by:

$$\text{Cloud-free metrics} = \frac{\text{Number of cloud-free images}}{\text{Total number of images}} \tag{5}$$

In the following results, agents trained with historical cloud data differ from agents trained with stochastic cloud data by the "data" suffix in the legends. Initially, the results for the binary reward model (shown in Equation (1)) are presented. Then, the results for the linear reward model (shown in Equation (2)) are discussed. An additional baseline case, NormalSat pre-filtered, was added for comparison in the binary reward model. The NormalSat pre-filtered has the same policy as the NormalSat, but only requests with a predicted cloud coverage of less than 0.2 are considered, representing the ground station filtering the requests before sending them to the satellite.

Binary reward model.

Agents performance. The agents with five different observation capabilities were tested in the cloud environment. The results for the CloudSat σ case were not included since the performance was similar to the CloudSat case. Figure 4 presents the reward for each agent across the number of requests (targets), in addition to the number of cloud-free and cloud-covered images and the cloud-free ratio. The reward indicates that the agents that observe c_f obtain better performance than those that cannot observe it, which is expected since it is the main parameter associated with cloud coverage. The agents that can observe σ in addition to c_f perform comparable or slightly better than cases that can observe only c_f; this result can be attributed to the fact that the standard deviation of the cloud coverage forecast is not as relevant as c_f in this scenario. Further, agents that cannot observe c_f nor σ show worse performance but are still superior to the NormalSat until 6,000 requests. When comparing agents trained with stochastic cloud data and historical data, the performance is similar in terms of reward, but the cloud-free ratio is higher for agents trained with historical data.

The agent with the worst performance in terms of reward is the NormalSat pre-filtered. This agent has a cloud-free ratio close to 1, but the number of cloud-free images is significantly lower than all other agents. This is due to the pre-filter applied by the ground station, which results in fewer requests being available to the satellite. Moreover, requests are pre-filtered based on the predicted cloud coverage, c_f, but can have a higher or lower true cloud coverage. In the case of NormalSat, requests with $c_f > 0.2$ are available and, once imaged, can still result in rewards if the true cloud coverage is less than 0.2 (which is the threshold considered for the binary reward model). In the case of the NormalSat pre-filtered, besides targets with $c_f > 0.2$ not being available, requests with $c_f < 0.2$ can still have a true cloud coverage, c_p, higher than 0.2, resulting in cloud-covered targets being imaged. A $c_f = 0.2$ filter was used to compare with the threshold used in the binary reward model, but other values can be used; tests were conducted with different values, and the results indicate that the performance increases with the threshold value until matching the NormalSat performance.

The different approach used by each agent is seen in Figure 5. Agents with the ability to observe the cloud coverage forecast prioritize targets with lower c_f, which results in a higher probability

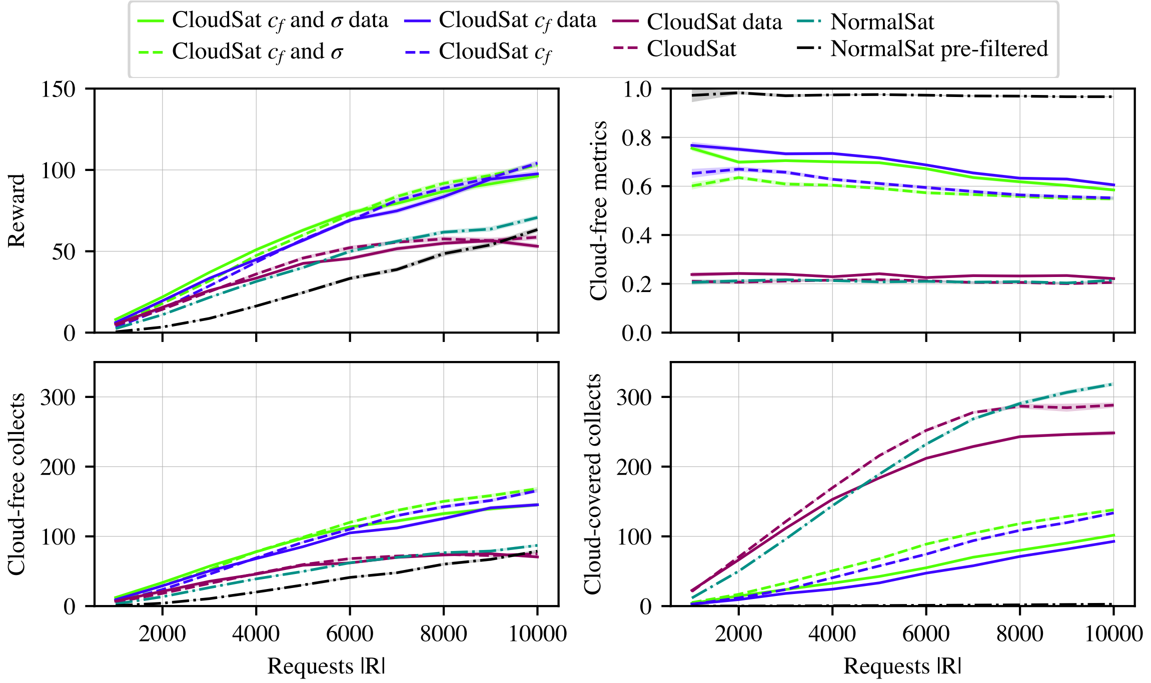

Figure 4. Reward, cloud-free metrics, and number of cloud-free and cloud-covered images across the number of requests for binary reward model.

of getting rewards. The two agents that can observe c_f and are trained with historical data tend to select targets with the lowest c_f among all cases (except NormalSat pre-filtered). In terms of priority, agents without knowledge of c_f select targets with higher priority, on average. This is seen mainly in the case of the NormalSat and NormalSat pre-filtered, which select targets with a priority of around 0.8 or higher. In combination with the information in Figure 4, it indicates that the NormalSat tries to select targets with higher priority to maximize the reward in sacrifice of the total number of imaged targets. In comparison, CloudSat and CloudSat data try to image more targets, sacrificing priority.

In an environment where clouds can occlude targets and no information about them is observed, maximizing the number of targets is more likely to provide better rewards. However, when the number of requests increases, CloudSat and CloudSat data miss more targets since they do not have time to settle and take the picture, as indicated in the missed metrics, leading to a worse performance than the NormalSat. Missed metrics relate the total number of missed targets to the total number of imaging attempts. NormalSat pre-filtered shows the lowest average c_f due to the pre-filtering applied by the ground station. Lastly, the σ parameter does not seem to impact the agent's performance significantly since the average value of selected targets' σ is the same as the average σ. This explains why the CloudSat σ agent performs similarly to the CloudSat agent.

Resources management. The reward metrics in Figure 4 indicate that the CloudSat and CloudSat data perform similarly to the NormalSat. However, Figure 6 shows that the NormalSat has a lower average battery level than all other agents. Hence, although performance is similar, agents trained in a cloud environment show better resource management even without observing the information. Part of these results can be explained by the NormalSat agent being more selective with the requests being tasked (priority), resulting in longer imaging actions and less time for charging actions. Nor-

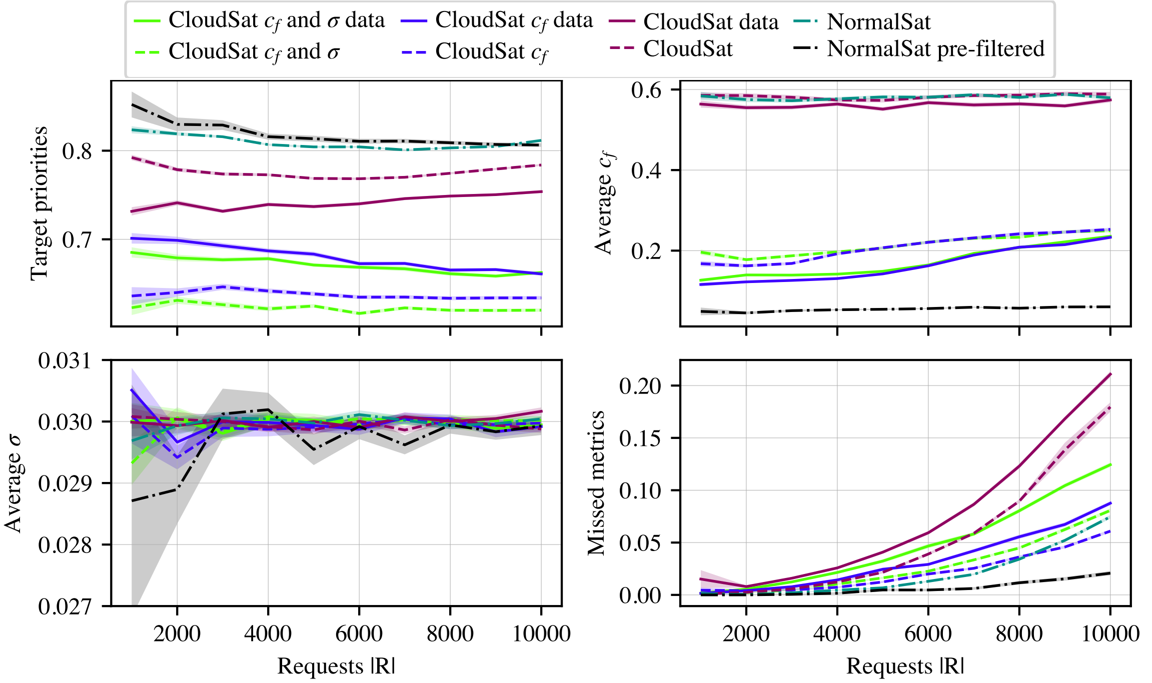

Figure 5. Information of agents' selected targets and missed metrics across the number of requests for binary reward model.

malSat pre-filtered shows similar battery levels to the NormalSat, indicating that the pre-filtering didn't improve its resource management. The resource management reflects the number of cases where the agents completed the three-orbit long episodes without dying; the NormalSat pre-filtered and NormalSat completed the three-orbit episodes in 86.2% and 95.9% of the cases, respectively, compared to more than 99.3% for the other agents.

Figure 6 also shows the average number of actions taken by each agent during each three-orbit-long episode and the ratio of imaging actions compared to total actions. CloudSat and CloudSat data agents take more actions than the NormalSat and have a high ratio of imaging actions, which is consistent with the higher number of imaged targets. CloudSat c_f and σ and CloudSat c_f have the lowest ratio of imaging actions, indicating that they leave more time for charging actions. The NormalSat pre-filtered has a lower number of actions than the NormalSat, which happens due to fewer requests being available to the satellite due to the filter. While NormalSat shows an intermediate number of actions compared to other agents, most of its actions are imaging actions, which is consistent with its worse resource management.

Linear reward model.

Agents performance. Different from the binary reward model, the cases trained with a linear reward model show less difference between agents that can and cannot observe cloud information as seen in Figure 7; this occurs due to the partial credit assigned based on cloud coverage and also the higher cloud threshold considered (0.7 in comparison to 0.2). Still, there is a more significant difference between agents trained with stochastic and historical cloud data. Surprisingly, the agents trained with historical data show worse performance in terms of reward; still, the cloud-free ratio is higher. The superior performance of the CloudSat c_f and σ and CloudSat c_f can be attributed to the

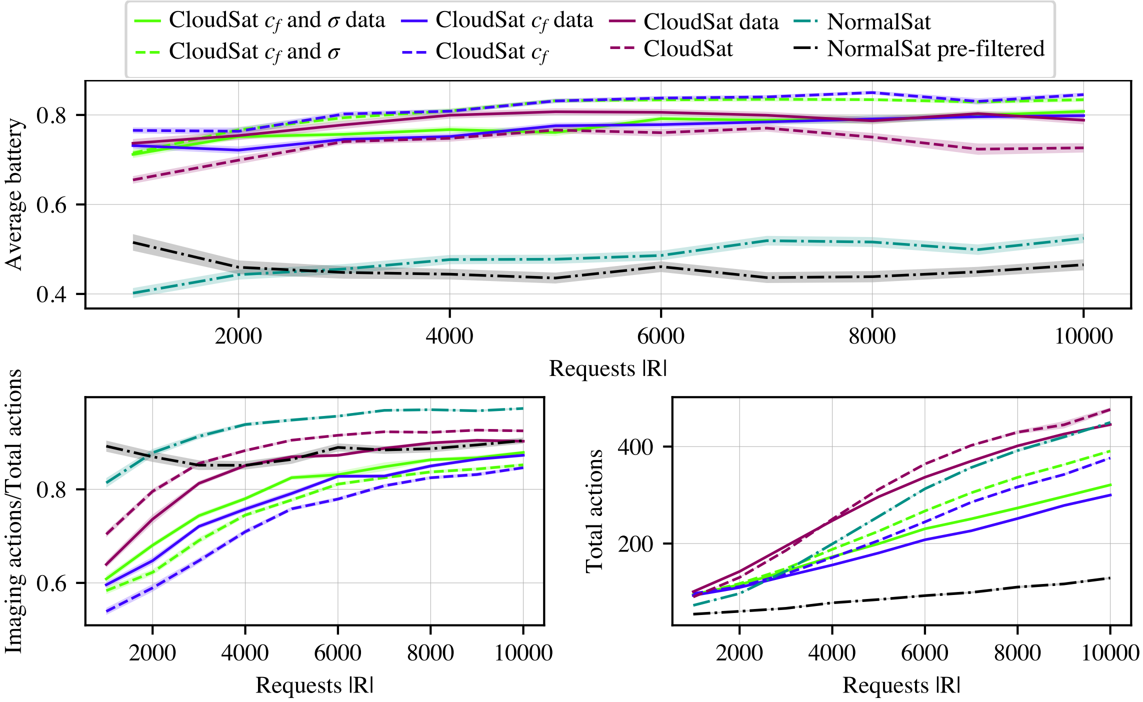

Figure 6. Average battery level for each agent across the number of requests for binary reward model.

agents' number of cloud-free images. The higher cloud-free metrics obtained by all agents in the linear reward model are explained by the fact that only targets with more than 0.7 cloud coverage are considered to be covered by clouds when compared to the threshold of 0.2 in the binary reward model.

Similar to the agents with binary reward model, Figure 8 shows that CloudSat c_f and σ and CloudSat c_f agents select targets with lower target priorities and higher cloud coverage, on average, than CloudSat c_f and σ data and CloudSat c_f data agents when using the linear reward model. Therefore, the agents trained with the stochastic cloud coverage are less selective with targets, allowing them to take more images, which leads to higher rewards. On the other hand, agents trained with data are more conservative. Results obtained for both the binary and linear reward models indicate that training the agent with the stochastic cloud model results in comparable or better performance in terms of rewards, which removes the need to store a large cloud dataset for training. Moreover, training with stochastic clouds leads to less spatial correlation between targets and more variability, which tends to be beneficial since it introduces the agent to more diverse scenarios.

Resources management. The average battery levels shown in Figure 9 indicate that the agent trained in an environment with clouds but deployed in the environment without clouds (NormalSat) still has the worst resource management. CloudSat c_f and σ data and CloudSat c_f data also show a lower battery level than the agents trained with stochastic cloud coverage. Still, agents trained with the stochastic cloud model show higher average battery levels. Similar to the binary reward model, the NormalSat agent successfully completed the three-orbit episodes in 95.8% of the cases, while other cases had a success rate of more than 99.5%. The total number of actions and the ratio of imaging actions for the linear reward model are similar to the binary case and are omitted for

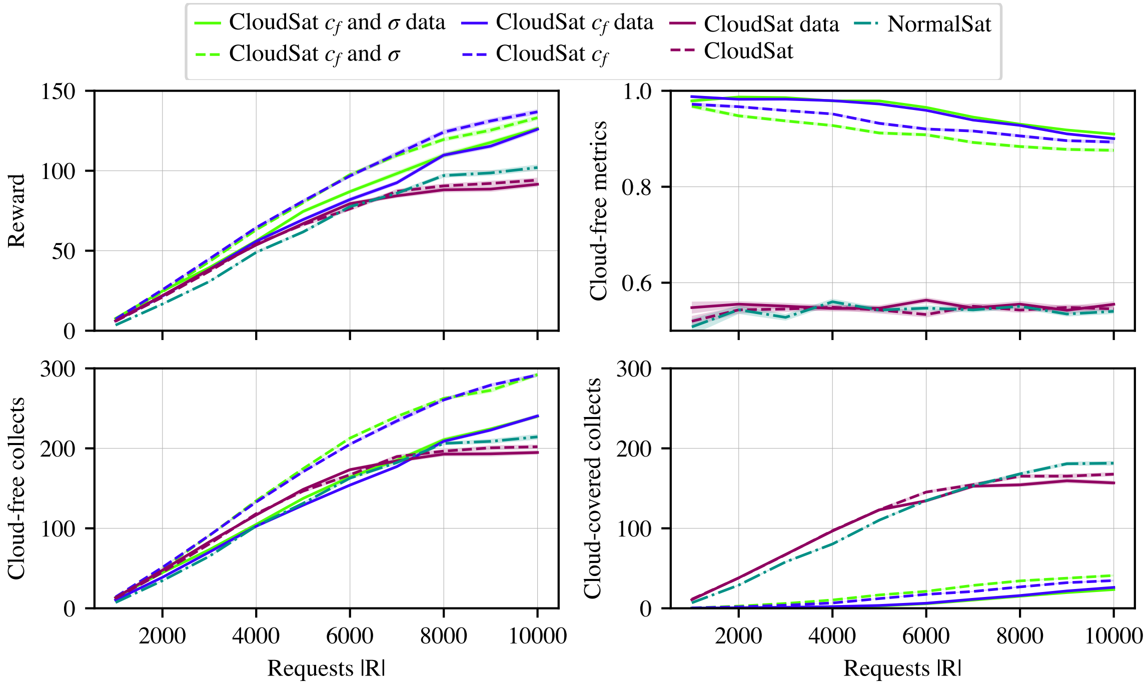

Figure 7. Reward, cloud-free metrics, and number of cloud-free and cloud-covered images across the number of requests for linear reward model.

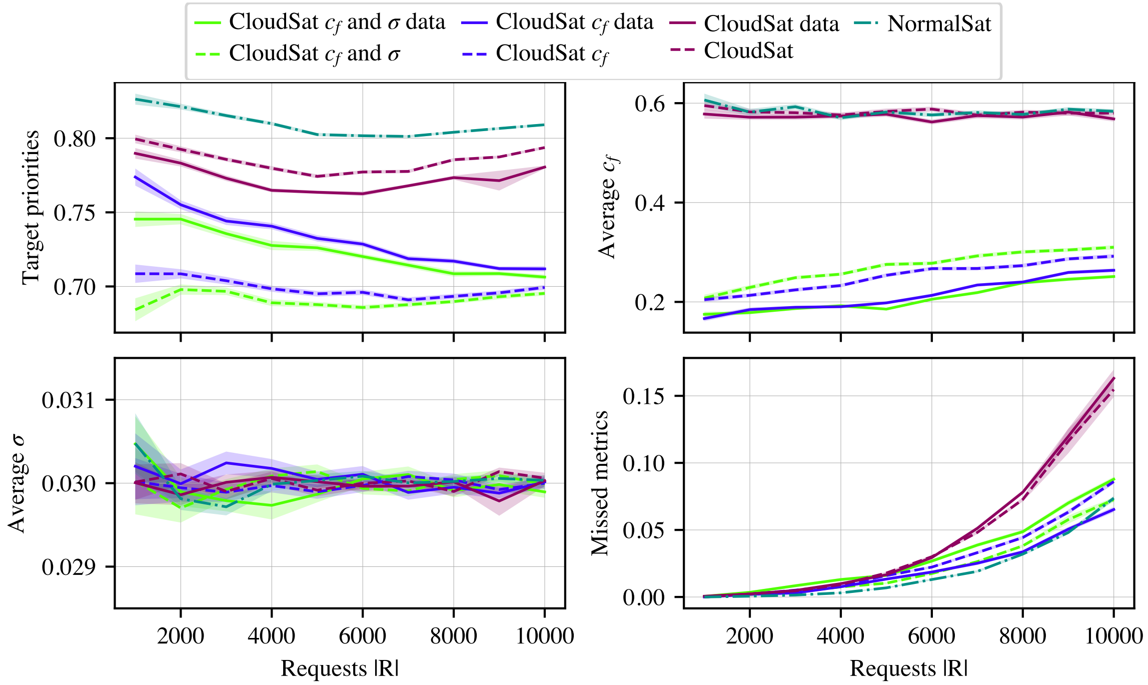

Figure 8. Information of agents' selected targets and missed metrics across the number of requests for linear reward model.

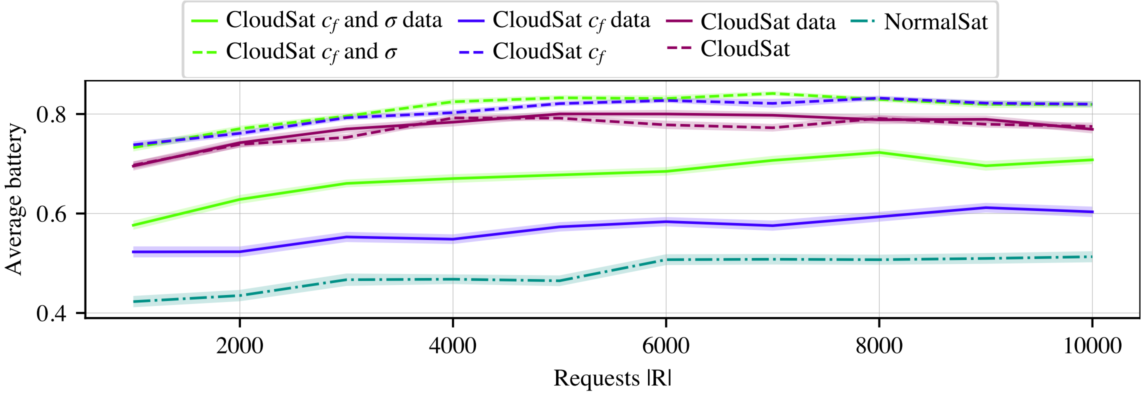

Figure 9. Average battery level for each agent across the number of requests for linear reward model.

brevity.

Comparative analysis.

The histogram in Figure 10 shows the distribution of cloud coverage probabilities of the targets imaged by the CloudSat c_f and σ with binary and linear reward models, helping to understand the different approaches used for each reward model. The agents with binary reward models tend to select targets with lower cloud coverage, while the agents with linear reward models are less selective with their targets. This behavior happens due to the difference in the threshold used for rewards and the partial credit assigned in the linear reward model. Such comparative analysis can be used to guide the selection of reward models and thresholds for the agent.

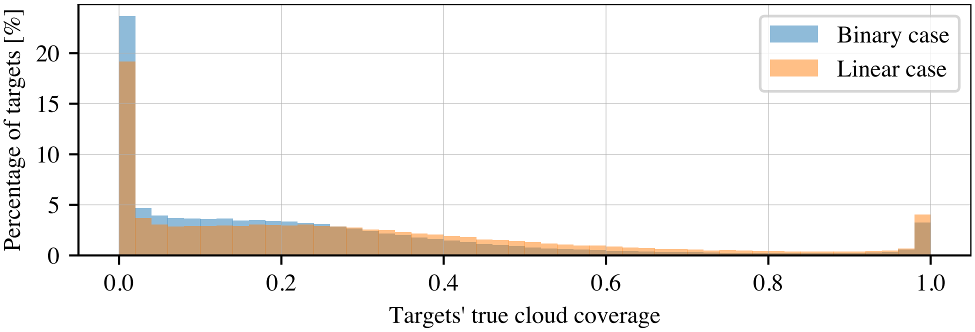

Figure 10. Histogram of cloud coverage of selected targets by the CloudSat c_f and σ with binary and linear reward models.

CONCLUSION

In this paper, the AEOS scheduling problem is formulated as a POMDP, and the use of DRL is proposed to solve the problem under cloud coverage. The results show that the agent's performance improves when observing the cloud coverage forecast, while the effect of the standard deviation did not show a significant impact. Interestingly, agents trained with stochastic cloud coverage show comparable or better performance than agents trained with historical cloud data, removing the need to store a large cloud dataset for training. Moreover, agents trained in an environment with clouds but incapable of observing cloud information show better resource management than agents trained in an environment without clouds; considering that clouds are unavoidable in the real world, agents training should consider the occlusion of clouds, even if they cannot directly observe the information.

Therefore, this study shows that the proposed DRL framework can be used to solve the AEOS problem under cloud coverage, considering on-board decision-making. Additionally, it can be combined with other approaches to provide cloud coverage forecast as input. Despite being focused on cloud coverage, the proposed framework can be adapted to consider other weather conditions with uncertainty affecting image quality, such as humidity.

Two distinct and commonly seen reward models were tested, showing similar results. In practice, the selection of reward models should be based on the needs of the satellite operator and other stakeholders. The results obtained in this study can be used to guide the selection of reward models and observation capabilities for the agent.

Future work should address the potential for satellite operators to prefer different reward models for different targets, such as a binary model and a linear model for another, each with a specific cloud coverage threshold. Therefore, having a single policy capable of accounting for different reward formats would be beneficial. Moreover, the proposed framework should be extended to consider the time-varying condition of clouds and evaluate the impact of outdated cloud information on the agent's performance. Additionally, future studies should address more constraints, such as storage capacity, and actions like downlink and reaction wheels momentum management, in addition to investigating a multi-satellite system to share information and improve the overall system performance.

ACKNOWLEDGMENT

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).

REFERENCES

[1] J. Ju and D. P. Roy, "The Availability of Cloud-free Landsat ETM+ Data Over the Conterminous United States and Globally," *Remote Sensing of Environment*, Vol. 112, Mar. 2008, pp. 1196–1211, 10.1016/j.rse.2007.08.011.

[2] 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, https://doi.org/10.1016/S1270-9638(02)01173-2.

[3] 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.

[4] 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.

[5] 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.

[6] X. Wang, G. Wu, L. Xing, and W. Pedrycz, "Agile Earth Observation Satellite Scheduling Over 20 Years: Formulations, Methods, and Future Directions," *IEEE Systems Journal*, Vol. 15, Sept. 2021, pp. 3881–3892, 10.1109/JSYST.2020.2997050.

[7] A. Herrmann, M. Stephenson, and H. Schaub, "Reinforcement Learning For Multi-Satellite Agile Earth Observing Scheduling Under Various Communication Assumptions," *AAS Guidance and Control Conference*, Breckenridge, CO, Feb. 2–8 2023. Paper No. AAS-23-146.

[8] M. Stephenson, L. Q. Mantovani, S. Phillips, and H. Schaub, "Using Enhanced Simulation Environments to Improve Reinforcement Learning for Long-Duration Satellite Autonomy," *AIAA Science and Technology Forum and Exposition (SciTech)*, Orlando, FL, Jan. 8–12 2024, 10.2514/6.2024-0990.

[9] A. Herrmann and H. Schaub, "Reinforcement Learning For Small Body Science Operations," *AAS Astrodynamics Specialist Conference*, Charlotte, NC, Aug. 7–10 2022. Paper No. AAS 22-563.

[10] 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.

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

[12] J. Wang, E. Demeulemeester, and D. Qiu, "A Pure Proactive Scheduling Algorithm for Multiple Earth Observation Satellites Under Uncertainties of Clouds," *Computers & Operations Research*, Vol. 74, Oct. 2016, pp. 1–13, 10.1016/j.cor.2016.04.014.

[13] J. Wang, E. Demeulemeester, X. Hu, D. Qiu, and J. Liu, "Exact and Heuristic Scheduling Algorithms for Multiple Earth Observation Satellites Under Uncertainties of Clouds," *IEEE Systems Journal*, Vol. 13, Sept. 2019, pp. 3556–3567, 10.1109/JSYST.2018.2874223.

[14] J. Wang, E. Demeulemeester, X. Hu, and G. Wu, "Expectation and SAA Models and Algorithms for Scheduling of Multiple Earth Observation Satellites Under the Impact of Clouds," *IEEE Systems Journal*, Vol. 14, Dec. 2020, pp. 5451–5462, 10.1109/JSYST.2019.2961236.

[15] C. G. Valicka, D. Garcia, A. Staid, J.-P. Watson, G. Hackebeil, S. Rathinam, and L. Ntaimo, "Mixed-integer Programming Models for Optimal Constellation Scheduling Given Cloud Cover Uncertainty," *European Journal of Operational Research*, Vol. 275, June 2019, pp. 431–445, 10.1016/j.ejor.2018.11.043.

[16] A. Hadj-Salah, R. Verdier, C. Caron, M. Picard, and M. Capelle, "Schedule Earth Observation Satellites with Deep Reinforcement Learning," Berkeley, USA, IWPSS, Nov. 2019.

[17] 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.

[18] 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.

[19] C. Zhang, L. Yuan, M. Xie, S. Zhang, and J. Li, "Autonomous Mission Planning of Earth Observation Satellite Based on Onboard Cloud Detection," *Advances in Space Research*, Vol. 70, Oct. 2022, pp. 2178–2194, 10.1016/j.asr.2022.07.007.

[20] Y. Gu, C. Han, Y. Chen, and W. W. Xing, "Mission Replanning for Multiple Agile Earth Observation Satellites Based on Cloud Coverage Forecasting," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 15, 2022, pp. 594–608, 10.1109/JSTARS.2021.3135529.

[21] C. Han, Y. Gu, G. Wu, and X. Wang, "Simulated Annealing-Based Heuristic for Multiple Agile Satellites Scheduling Under Cloud Coverage Uncertainty," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 53, May 2023, pp. 2863–2874, 10.1109/TSMC.2022.3220534.

[22] A. Candela, J. Swope, and S. A. Chien, "Dynamic Targeting to Improve Earth Science Missions," *Journal of Aerospace Information Systems*, Vol. 20, Nov. 2023, pp. 679–689, 10.2514/1.I011233.

[23] K. Naik, O. Chang, and C. Kotulak, "Deep Reinforcement Learning for Autonomous Satellite Responsiveness to Observed Events," *2024 IEEE Aerospace Conference*, Big Sky, MT, USA, IEEE, Mar. 2024, pp. 1–10, 10.1109/AERO58975.2024.10521008.

[24] 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.

[25] H. Hersbach, B. Bell, P. Berrisford, G. Biavati, A. Horányi, Muñoz-Sabater, J. Nicolas, C. Peubey, R. Radu, I. Rozum, D. Schepers, A. Simmons, C. Soci, D. Dee, and J. Thépaut, "ERA5 Hourly Data on Single Levels from 1940 to Present," *Copernicus Climate Change Service (C3S) Climate Data Store (CDS)*, 2023, 10.24381/cds.adbb2d47.

[26] M. J. Kochenderfer, T. A. Wheeler, and K. H. Wray, *Algorithms for decision making*. Cambridge, Massachusetts: The MIT Press, 2022.

[27] M. A. Stephenson and H. Schaub, “BSK-RL: Modular, High-Fidelity Reinforcement Learning Environments for Spacecraft Tasking,” *75th International Astronautical Congress*, Milan, Italy, IAF, Oct. 2024.

[28] M. Stephenson, L. Q. Mantovani, and H. Schaub, “Intent Sharing For Emergent Collaboration In Autonomous Earth Observing Constellations,” *AAS Guidance and Control Conference*, Breckenridge, CO, Feb. 2–7 2024. Paper No. AAS 24-192.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” Aug. 2017. arXiv:1707.06347 [cs].

[30] University of Colorado Boulder Research Computing, “Alpine,” 2023. Publisher: University of Colorado Boulder, 10.25811/K3W6-PK81.

[31] 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.

[32] A. Harris and H. Schaub, “Spacecraft Command and Control with Safety Guarantees using Shielded Deep Reinforcement Learning,” *AIAA SciTech*, Orlando, Florida, Jan. 6–10 2020.

[33] I. Nazmy, A. Harris, M. Lahijanian, and H. Schaub, “Shielded Deep Reinforcement Learning for Multi-Sensor Spacecraft Imaging,” *American Control Conference*, Atlanta, Georgia, June 8–10 2022.

[34] R. Reed, H. Schaub, and M. Lahijanian, “Shielded Deep Reinforcement Learning for Complex Spacecraft Specifications,” *American Control Conference*, Toronto, Canada, July 8–12 2024.