Shielded Deep Reinforcement Learning for Multi-Sensor Spacecraft Imaging

Islam Nazmy[1], Andrew Harris[1], Morteza Lahijanian[2], and Hanspeter Schaub[2]

Abstract— **This paper considers the problem of autonomous spacecraft control for imaging missions, subject to safety constraints. The controller chooses between discrete flight modes to image a target with different sensor types. The safety constraints include maintaining safe battery levels, reaction wheel speeds, and body rates. The proposed approach applies shielded deep reinforcement learning (SDRL) to autonomously command spacecraft flight modes, where the imaging requirements are communicated to the agent through a finite-state machine (FSM). The training is done in a target- and orbit-agnostic manner to create a single artificial neural network that can operate in a range of conditions. The FSM specifies which sensor type the agent should use for the next image. Simulation results based on a spacecraft tasked on Boulder, CO, USA demonstrate that this approach is effective for commanding a spacecraft safely while meeting predefined imaging requirements. This work also demonstrates how an agent trained on Boulder is capable of being applied to other Earth-targets as well as targets on the Moon with similar performance.**

I. INTRODUCTION

Autonomy in spacecraft operations has the potential to reduce operational costs and to make spacecraft more resilient to failures [1]. Onboard autonomy is difficult due to the high-dimensionality of spacecraft states and limited computation capabilities. Traditional optimization methods for this problem suffer from state explosion, while machine learning (ML) approaches lack safety guarantees. This work proposes using shielded deep reinforcement learning (SDRL) to avoid the limitations of traditional optimization while providing safety guarantees for spacecraft autonomy.

Traditional optimization techniques can solve spacecraft mode planning problems, however, the optimal solutions are often brittle to initial conditions. JPL's ASPEN system tasks spacecraft with Earth-imaging activity plans based on local search algorithms. When anomalies are encountered, solutions must be recomputed which can be computationally-intensive [2]. Work [3] propose using a maximal independent set algorithm to schedule non-conflicting actions for the spacecraft. Although this approach guarantees safety constraints, it must be recomputed when anomalies are encountered.

Deep reinforcement learning (DRL) approaches effectively handle the state-explosion problem and are robust to initial conditions [4], [5]. A Monte-Carlo Tree Search (MCTS) algorithm can solve the satellite activity scheduling problem, performing almost as well as traditional optimization techniques at a fraction of the execution time [6]. The risk of using DRL is that it does not provide safety guarantees. Work [7] applies an RL algorithm for imaging targets on an asteroid, but the agent repeatedly crashes into the asteroid to exploit the reward function.

Recent work in [8] demonstrates that applying a safety shield to DRL agents provides safety margins during operations. Furthermore, SDRL is shown to improve the policy performance and reduce training time for spacecraft operations [9]. This work applies SDRL to the Earth-imaging problem where the spacecraft should take images according to some imaging criteria. A deterministic policy is explored to shield the ML agent from taking unsafe actions that violate the safety constraints. The shielded agent is trained on an Earth-imaging simulation, where the agent is rewarded for satisfying some predefined imaging criteria.

The novel contribution of this work is applying SDRL to the spacecraft imaging problem. The performance of the agent to maintain spacecraft safety states and switch the image type according to predefined imaging requirements is studied through high-fidelity numerical simulation. Furthermore, the study investigates how to train an agent to be both location-agnostic and planet-agnostic. This general solution can be applied to different imaging scenarios without requiring additional training. With the proposed approach an agent trained on a single Earth target can be used on another Earth target or around another celestial object like the Moon to observe a Lunar surface target.

II. BACKGROUND

A. *Markov Decision Process*

Sequential decision problems can be formally posed as a Markov Decision Process (MDP), where an agent chooses an action based on observing a state and then receives a reward signal corresponding to the state-action pair [10]. The solution to an MDP is an optimal policy which selects actions to maximize the cumulative reward of the decision process.

An MDP is a tuple ($\mathcal{S}$, $\mathcal{A}$, T, R, γ), where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the the action space, T is the transition probability function such that $T(s'|s,a)$ is the probability of transitioning to state s' from state s with action a, R is the reward function such that $R(s,a,s')$ represents the reward signal from transitioning into state s' from state s with

This work was supported in part by the Air Force STTR award FA8649-21-P-0137 and University of Colorado Boulder.

[1] I. Nazmy and A. Harris are research assistants with the Smead Aerospace Engineering Sciences Department, University of Colorado at Boulder, Boulder, CO 80303. email:{islam.nazmy, andrew.harris}@colorado.edu

[2] M. Lahijanian and H. Schaub are faculty with the Smead Aerospace Engineering Sciences Department, University of Colorado at Boulder, Boulder, CO 80303. email:{morteza.lahijanian, hanspeter.schaub}@colorado.edu

action a, and finally γ is the discount factor, which imposes the significance of future rewards compared to immediate rewards. The discount factor also makes the cumulative expected reward finite for infinite-horizon problems. The solution to an MDP is a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$, which maps states to actions.

B. Shielded Deep Reinforcement Learning

SDRL is a framework for applying RL algorithms to solve MDPs while constrained by safety bounds. Shielded learning techniques use a safety MDP to limit unsafe actions from an agent with respect to safety requirements [8]. Each state is a coarse discretization of the spacecraft states bounded by safety limits. Examples of unsafe states include low battery, high momentum wheel speeds, or high tumbling rate.

A simple safety MDP is shown in Fig. 1 where s_0 is a safe state, s_1 is an unsafe state, s_f is the failure state, and the action space is $\mathcal{A} = \{a_0, a_1\}$. The states can transition probabilistically; in Fig. 1 action a_0 in state s_1 can transition to s_1 or s_f. The transition probabilities, $\mathcal{T}(s'|s,a)$, are selected when constructing the safety MDP.

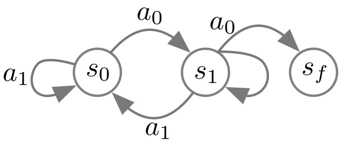


Fig. 1: Sample safety MDP with initial state s_0 and terminal state s_f.

The safety MDP can be solved by formal synthesis [11], [12] techniques using temporal logic to define the safety requirements. Linear temporal logic (LTL) can be used to mathematically describe the requirement of the final policy [13]. With the example of Fig. 1, the LTL specification of this safety game can be simply expressed as (1) which reads, “always avoid the unsafe state s_f”. The solution to this safety game is used as the shield policy in an SDRL framework.

$$\varphi = G(\neg s_f) \tag{1}$$

A post-posed shielded learning framework is shown in Fig. 2, where the shield checks that the agent’s action aligns with the safety MDP policy. If the agent’s action violates the shield policy, the shield overrides it with the safe action.

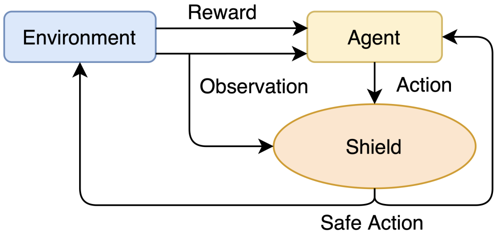


Fig. 2: Shielded reinforcement learning agent architecture [9]

III. PROBLEM FORMULATION

The problem posed is an autonomous form of the Earth-observing satellite scheduling and planning problem [14]. For a spacecraft in Earth-orbit, an agent is to select a sequence of flight modes such that the spacecraft meets some imaging requirements while maintaining the spacecraft’s safety. The action space $\mathcal{A} = \{a_i\}_{i=0}^{3}$ is the following set of spacecraft modes: *Imaging Mode A* (a_0); *Imaging Mode B* (a_1); *Charging Mode* (a_2); and *Momentum Dumping Mode* (a_3). The imaging modes command a nadir-pointing orientation which aligns the spacecraft body with the Hill frame, pointing the camera boresight Nadir. An MRP-feedback law from Chapter 8 in [15] is used to control the spacecraft’s reaction wheels. Both imaging modes are assumed to have the same boresight direction, and thus the same orientation.

The *Charging Mode* commands the spacecraft to point its solar panels in the sun-direction, also using an MRP-feedback law. The *Momentum Dumping Mode* commands the spacecraft to dump reaction wheel momentum greater than a minimum threshold; this mode is also referred to as reaction wheel desaturation. The spacecraft uses onboard thrusters to dump momentum if necessary. In *Momentum Dumping Mode*, the spacecraft is oriented such that the solar panels point away from the sun to differentiate the action from the *Charging Mode*.

An image of the target is successfully taken if the spacecraft is in an imaging mode while within the viewing cone of the target. The minimum elevation viewing angle of the target is el_{min}. Although the spacecraft is in each imaging mode for a continuous time interval, an image is only considered taken once at the start of the mode. The time elapsed since the last image of a target was taken is t_A for image A and t_B for image B. The counters are set to zero when an image of the target has been captured. The image counters can be formally defined in terms of the continuous step-size, t_{step}, as

$$t_A = \begin{cases} t_{\text{step}}, & el_{\text{sc}} > el_{\text{min}} \text{ and } a = a_0 \\ t_A + t_{\text{step}}, & \text{otherwise} \end{cases} \tag{2}$$

$$t_B = \begin{cases} t_{\text{step}}, & el_{\text{sc}} > el_{\text{min}} \text{ and } a = a_1 \\ t_B + t_{\text{step}}, & \text{otherwise} \end{cases} \tag{3}$$

The data storage of an image is not considered, however, imaging modes draw twice as much power from the spacecraft compared to the nominal operational power required.

The imaging requirements for the problem are time-dependent; a certain amount of time T_A and T_B should pass between taking image A and B, respectively. For this problem, $T_A \leq T_B$, and image type A should be preferred over B. The preferred image type can be formalized in terms of the time elapsed since taking image A and B.

$$a_{\text{preferred}} \in \begin{cases} \{a_0\}, & t_A \geq T_A \\ \{a_1\}, & t_B \geq T_B \text{ and } t_A < T_A \\ \{a_2, a_3\}, & \text{otherwise} \end{cases} \tag{4}$$

The agent is considered to have met the imaging requirement if $a_i = a_{\text{preferred}}$.

IV. APPROACH

This work applies an SDRL learning framework to the spacecraft imaging problem, shielding the spacecraft's power level, attitude rate, and reaction wheel spin rate. The safety states are defined in Table I in terms of the spacecraft stored charge p, attitude rate $|\dot{\sigma}|$, and wheel speed Ω. The shield policy that solves this safety game can be found in [9].

TABLE I: Safety Shield States

Unsafe State	State Boundary
Low Power	$p < 20\%$
Saturated Wheel	$\Omega > 0.7\Omega_{\text{max}}$
Uncontrolled Tumble	$\lvert\dot{\sigma}\rvert > 0.01$ rad/s

The reward function $R(s, a, s')$ is formulated such that the agent receives a maximum cumulative reward of 1 at the end of an episode if it takes the correct image type with perfect pointing accuracy at every imaging opportunity. The spacecraft receives a penalty of -1 if it encounters a failure. This can occur from the reaction wheels spinning more than the maximum rate or the battery draining completely. These failure modes can be generalized with the failure state s_f. The reward function can be formally expressed as

$$R(s, a, s') = \begin{cases} \frac{f}{N} \cdot \frac{1}{1+\epsilon_{\text{att}}^2}, & el_{\text{sc}} > el_{\text{min}} \text{ and } a \in \{a_0, a_1\} \\ -1, & s = s_f \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where N is the total number of steps that the spacecraft is within the viewing cone of the target, ϵ_{att} is the attitude error, and f is a factor corresponding to the image type. This factor f is defined as

$$f = \begin{cases} 1.0 & a_i = a_{\text{preferred}} \\ 0.1 & \text{otherwise} \end{cases} \tag{6}$$

The desired image type following the requirements T_A and T_B can be generically defined by a Finite State Machine (FSM), as shown in Fig. 3. The use of an FSM generalizes this approach to more than two image types; rather than following a complex set of if-else statements, reactive synthesis methods can generate an FSM corresponding to requirements written in LTL [13].

The discount factor of the MDP is $\gamma = 0.99$. Although the discount factor can be set to 1.0 for this finite-horizon problem, a discount factor less than one helps converge the value function when the problem is not perfectly Markovian. When states far in the future are more difficult to predict, a smaller discount factor makes those states in the far future negligible. Following a dimensionality reduction, there will inherently be some non-stationarity in the problem.

V. DIMENSIONALITY REDUCTION

One of the problems in applying RL algorithms to the real world is selecting an appropriate state space. The true state space of a spacecraft could include thousands of states when including every subsystem, however, not every feature contains useful information for a particular task. This is called the curse of dimensionality [16]. Reducing the dimensionality of the problem can be challenging; if the dimensionality of the problem is too high, no meaningful solution can be found, and if the dimensionality is too low, the model can become overfit to the data.

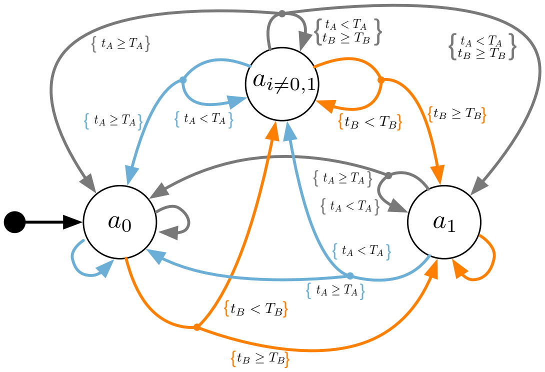


Fig. 3: FSM that defines the preferred image type, $a_{\text{preferred}}$. The orange transitions represent taking image A of the target, blue transitions represent taking image B of the target, and gray transitions indicate any other action taken. The transition guards in curly brackets indicate a condition that must be met for transition.

Another constraint on selecting the state space for RL algorithms is that the problem must remain Markovian; that is, the transition probabilities are independent of previous states, given the present state [17], [18]. This is a significant challenge for high-fidelity simulators, since a reduction of the simulation states will almost guarantee some non-stationarity to the problem. In reducing the dimensionality of the problem, the required fidelity of the solution and implementation period must be considered. This difference between the problem representation and the true problem is also known as the simulation gap.

Reducing the MDP state space to a subset of the simulation states is necessary to avoid the curse of dimensionality. Starting from the reward function in (5)-(6), the state space is composed of contributing states to the reward function such as Access Indicator, Spacecraft Mode, and Attitude Error. These states alone are not Markovian; for instance, the target access indicator at the next time-step cannot be predicted only with the current access indicator. Thus, the state space is expanded further to include states which make the state space Markovian. Some states depend on themselves at the previous time step, such as the counters t_A and t_B, which is indicated by a self-pointing arrow. The final network of the observation space is illustrated in Fig. 4 to show the correlation of states to the reward function.

Note that the selection of these states using expert knowledge can help reduce the dimensionality; for example, the spacecraft's relative position and velocity to the target are just as sufficient to calculate the spacecraft's access indicator at the next time-step as would the inertial position and

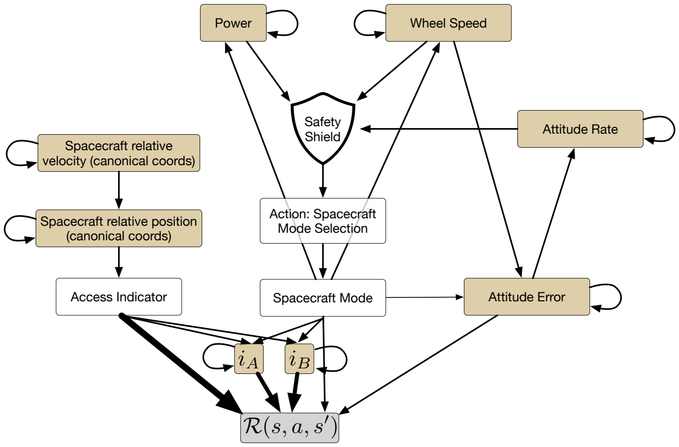

Power
Wheel Speed
Safety Shield
Attitude Rate
Spacecraft relative velocity (canonical coords)
Spacecraft relative position (canonical coords)
Action: Spacecraft Mode Selection
Access Indicator
Spacecraft Mode
Attitude Error
i_A
i_B
$\mathcal{R}(s, a, s')$

Fig. 4: Correlation graph of simulation states that affect the reward function. States which are approximately dependent on themselves are highlighted in gold. The approximate importance is shown by arrow weights.

velocity of both the spacecraft and the target, however, relative positions require only six states rather than twelve. Note that the relative position and velocity only give a first-order approximation.

Using the relative position and velocity of the spacecraft to the target also makes the problem location-agnostic; an agent trained with relative coordinates makes the policy independent of any particular target location. By reducing the dimensionality this way, a solution to one target location should also work sufficiently for any other Earth-target without additional training.

Similarly, the spacecraft's position and velocity may be expressed in canonical coordinates which normalize the trajectory to planets in a two-body regime [15]. The absolute position $\mathbf{r}$ and velocity $\mathbf{v}$ are transformed to canonical coordinates with the transformation in (7)-(8).

$$\mathbf{r}_{\text{can}} = \frac{\mathbf{r}}{r_{\text{eq}}} \tag{7}$$

$$\mathbf{v}_{\text{can}} = \frac{\mathbf{v}}{\sqrt{\mu/r_{\text{eq}}}} \tag{8}$$

The state space of the agent is 12 dimensional, which includes: spacecraft relative position and velocity to the target in the SEZ frame, expressed in canonical units; MRP attitude error 2-norm (ϵ_{att}); attitude rate 2-norm; maximum wheel speed, normalized to the wheel limit; stored charge, normalized to the battery limit; minutes elapsed since last image A of the target; minute elapsed since the last image B of the target. Note that scaling the parameters to an order of 1 helps with training due to layer normalization [19].

VI. NUMERICAL SIMULATIONS

A. Spacecraft Environment

The spacecraft is simulated using the Basilisk Astrodynamics Software Framework, a high-fidelity spacecraft simulator [20]. The spacecraft's selected orbit has almost exactly 30 minutes of viewing time over Boulder, CO, USA, and is never in eclipse. The full orbital parameters are shown in Table II. In addition to the two-body gravitational force, the spacecraft experiences a disturbance torque of 0.2 mN·m in a random direction. The dynamics are propagated with 0.5 s integration steps.

TABLE II: Earth Orbit Parameters

a	13,878 km
e	0.00001
i	53.0°
Ω	115.0°
ω	5.0°
f	240.0°
Epoch	2021 May 04 06:47:49 (UTC)

The spacecraft power and attitude subsystems are modelled in Basilisk with the parameters in Table III. The momentum wheels draw additional power according to the wheel speeds, and the imaging modes draw an extra 5W of power when active. The spacecraft is equipped with three momentum wheels and three thrusters along the body axes. The wheel speeds, tumble rate, and battery level are initialized randomly at the start of each episode.

TABLE III: Spacecraft Attributes

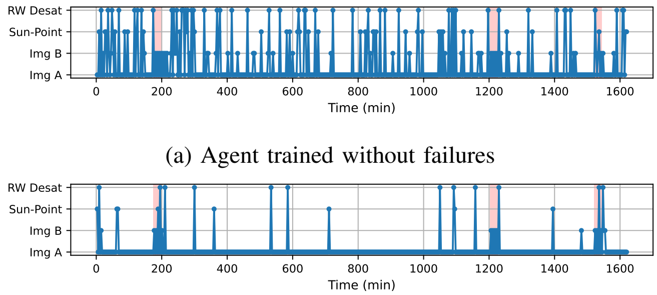
Mass	330.0 kg
Hub Dimensions	1.38 m × 1.04 m × 1.58 m
Wheel Limit	3000 rpm
Battery Capacity	50.0 W·Hr
Nominal Power Draw	5W
Solar Panel Area	0.09 m^2
Solar Cell Efficiency	0.20

Each flight-mode action is executed with three-minute intervals, where the attitude feedback control is executed with 1.0 s time-steps.

Although the agent is not rewarded for taking images outside the Boulder viewing cone, it is not penalized for excess images. Initial results showed the agent switching modes frequently which made it difficult to interpret the agent's strategy. To reduce the mode switching frequency, the training environment imposes a stochastic failure model when the agent attempts to switch to a different mode. This failure model is meant to encompass the risks associated with frequent mode switching, and also make the agent's strategy easier to interpret. Fig. 5 shows the spacecraft's action history for an agent trained with mode switching failures and an agent trained without failures.

B. Training Parameters

The Basilisk simulator is used to create a custom `gym` environment for use with the `stable-baselines` implementation of PPO2 [21]. The agent is trained with a learning rate of $\alpha = 0.00025$. The inner layers of the fully-connected network are of dimension [96,96], with a hyperbolic tangent activation function. The network is trained for 5×10^6 time-steps. The network is trained on an AMD



(a) Agent trained without failures

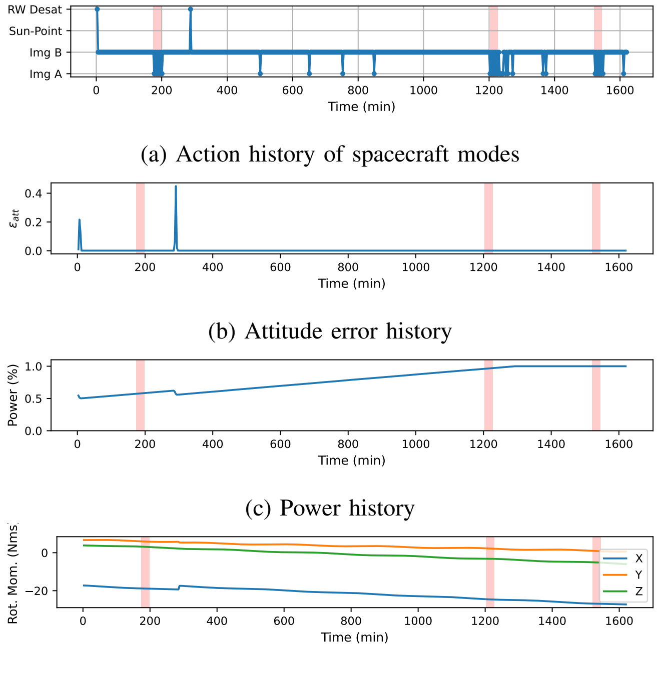
(b) Agent trained with 1% failure probability

Fig. 5: Action history of agent trained with and without mode-switching failures.

Ryzen Threadripper™3960X 24-Core processor @3.8GHz with 64 GB of memory.

C. Location-Agnostic Agent

An SDRL agent trained on a target at Boulder, CO is shown operating safely in Fig. 6. The spacecraft starts with a high attitude error because of the randomly initialized attitude, but maintains a low attitude error for most of the episode while performing some momentum dumping maneuvers to counter the external torque. Since the spacecraft is in sunlight for the entire simulation, it does not need to stay in sun-pointing mode to keep the battery charged. The momentum dumping mode does require the spacecraft to turn away from the sun, showing that the agent makes low-risk trade-offs when necessary.



(a) Action history of spacecraft modes

(b) Attitude error history

(c) Power history

(d) Rotational Angular Momentum history

Fig. 6: Agent observing target at Boulder, CO, USA. Red regions indicate when the spacecraft is in the visibility cone.

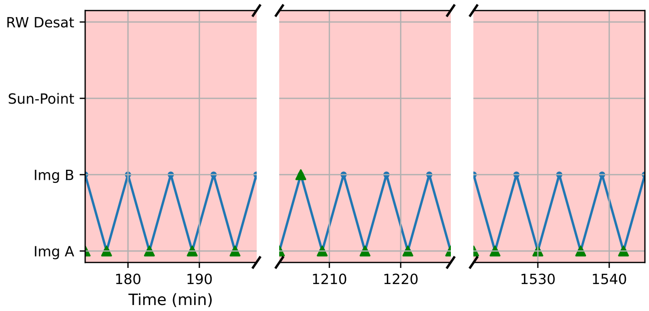


Fig. 7: Imaging closeup of spacecraft observing target at Boulder, CO, USA. Green markers indicate the desired image type by the FSM.

The spacecraft switches between imaging modes inside the viewing cone, as highlighted in Fig. 7. This example shows the agent almost always matching the image type of the FSM. In the first and third passes of Fig. 7, the agent captures image B before Image A; although this does not match the imaging requirements exactly (prioritizing Image A before Image B), the agent still has time to capture Image A at least once.

The Boulder-trained agent also performs well when imaging a target on the opposite side of the Earth from Boulder, in the Indian Ocean. Fig. 8 shows the agent taking the appropriate image type over the new target, with a similar imaging behavior to how it performs over Boulder.

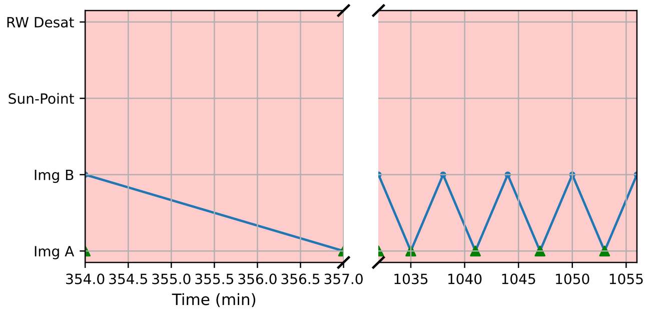


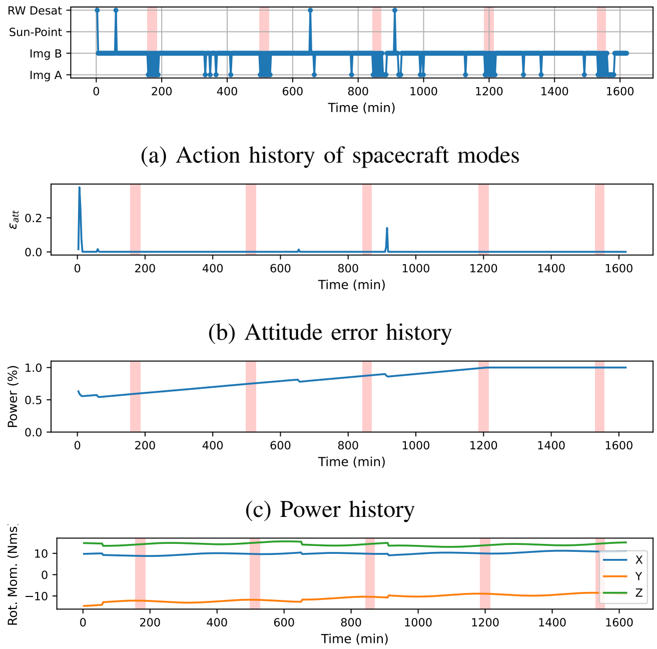
Fig. 8: Imaging closeup of spacecraft observing target in the Indian Ocean.

Without any changes to the agent architecture, the spacecraft still performs well on the Indian Ocean target although it was trained on Boulder. The selection of target-relative coordinates makes the spacecraft agnostic to the target it is imaging.

D. Planet-Agnostic Agent

The same agent trained on Boulder was also applied to a Lunar mission. The orbit selected has a 2000 km altitude, otherwise the same orbit parameters from Table II. Note that the dynamics only simulate two-body effects.

The spacecraft still performs safely around the Moon as shown in Fig. 9, dumping angular momentum when



(a) Action history of spacecraft modes

(b) Attitude error history

(c) Power history

(d) Rotational Angular Momentum history

Fig. 9: Agent observing target on the Moon from Lunar orbit.

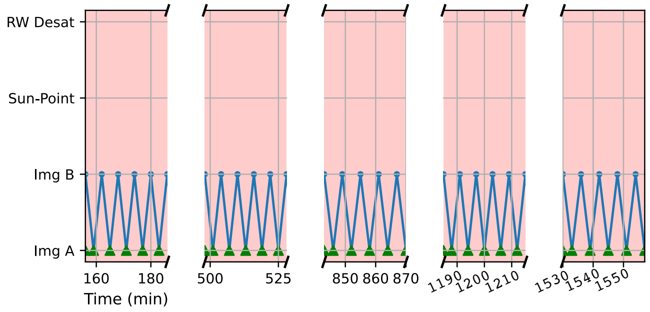


Fig. 10: Closeup of spacecraft observing Lunar target.

necessary and managing the attitude error. The imaging behavior is still able to meet the FSM image requirements, shown in Fig. 10.

The selection of canonical coordinates for the state space allows the agent trained on Earth to be applied to another two-body system easily, given that it has similar environment characteristics to Earth. For planets further from Earth, the solar panels and reaction wheels should be resized according to the sunlight available and external torque.

VII. CONCLUSION

This work demonstrates a shielded deep reinforcement learning approach (SDRL) to solving the autonomous Earth observing satellite commanding. Earth locations are successfully imaged while satisfying safety constraints. One of the limitations of this approach is that the agent only learns the imaging behavior it was trained on; if the times T_A and T_B need to be adjusted, the agent needs to be retrained. Using SDRL for spacecraft autonomy has a significant advantage over rule-based autonomy or traditional optimization; this proposed method allows for agents to learn a behavior from a restricted set of initial conditions, and still operate successfully in environments it has not been trained on. Numerical simulations demonstrate that the neural network can image targets it was not trained on, as well as operate about other celestial bodies.

REFERENCES

[1] C. R. Frost, “Challenges and opportunities for autonomous systems in space,” in *Frontiers of Engineering: Reports on Leading-Edge Engineering from the 2010 Symposium*, 2010.

[2] R. Sherwood, A. Govindjee, D. Yan, G. Rabideau, S. Chien, and A. Fukunaga, “Using aspen to automate eo-1 activity planning,” in *1998 IEEE Aerospace Conference Proceedings (Cat. No. 98TH8339)*, vol. 3. IEEE, 1998, pp. 145–152.

[3] D. Eddy and M. J. Kochenderfer, “A maximum independent set method for scheduling earth-observing satellite constellations,” *Journal of Spacecraft and Rockets*, 2021.

[4] M. Nazari, A. Oroojlooy, L. V. Snyder, and M. Takáč, “Reinforcement learning for solving the vehicle routing problem,” *arXiv preprint arXiv:1802.04240*, 2018.

[5] Y. Li, N. Li, H. E. Tseng, A. Girard, D. Filev, and I. Kolmanovsky, “Safe reinforcement learning using robust action governor,” in *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, vol. 144. PMLR, 07 – 08 June 2021, pp. 1093–1104.

[6] A. Herrmann and H. Schaub, “Monte carlo tree search methods for the earth-observing satellite scheduling problem,” in *Journal of Aerospace Information Systems*. American Institute of Aeronautics and Astronautics, 2021.

[7] D. M. Chan and A.-A. Agha-Mohammadi, “Autonomous imaging and mapping of small bodies using deep reinforcement learning,” in *2019 IEEE Aerospace Conference*, Big Sky, MT, USA, 2019, pp. 1–12.

[8] M. Alshiekh, R. Bloem, and e. a. Ehlers, “Safe reinforcement learning via shielding,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[9] A. Harris and H. Schaub, “Spacecraft command and control with safety guarantees using shielded deep reinforcement learning,” in *AIAA SciTech Forum*, Jan 2020.

[10] M. J. Kochenderfer, *Decision making under uncertainty: theory and application*. MIT press, 2015.

[11] H. Kress-Gazit, M. Lahijanian, and V. Raman, “Synthesis for robots: Guarantees and feedback for robot behavior,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 211–236, May 2018.

[12] M. Lahijanian, S. B. Andersson, and C. Belta, “Formal verification and synthesis for discrete-time stochastic systems,” *IEEE Transactions on Automatic Control*, vol. 60, no. 8, pp. 2031–2045, Aug. 2015.

[13] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[14] 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*, 2020.

[15] H. Schaub and J. Junkins, *Analytical Mechanics of Space Systems*, 4th ed. AIAA Education, April 2018.

[16] M. Verleysen and D. François, “The curse of dimensionality in data mining and time series prediction,” in *International work-conference on artificial neural networks*. Springer, 2005, pp. 758–770.

[17] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[18] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[19] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2016.

[20] P. W. Kenneally, S. Piggott, and H. Schaub, “Basilisk: A flexible, scalable and modular astrodynamics simulation framework,” *Journal of Aerospace Information Systems*, vol. 17, no. 9, pp. 496–507, 2020.

[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.