TRUSTWORTHY REINFORCEMENT LEARNING FOR DECENTRALIZED CONTROL OF SATELLITES

Victor Bajenaru*, Adam Herrmann†, Hanspeter Schaub†, Jaime Ramirez*, Sean Phillips‡

There is a growing need for efficient onboard satellite Command and Control (C2) for both DoD and commercial purposes. We study multi-satellite decentralized onboard C2, comparing a Deep Reinforcement Learning (DRL) control policy with a rule-based policy. We consider hundreds of dynamic prioritized imaging requests from dispersed tactical requestors, routed to many satellites, and with periodic ground station coordination. We simulate spacecraft actions including: momentum wheel attitude control, momentum wheel dumping actions, pointing at the sun for battery recharge, and battery discharge as a result of momentum wheel actions and imaging. Compared to a rule-based policy, the DRL policy shows slightly better request fulfillment performance and slightly worse computation time. Once the simulation environment is created, we find the DRL policy is able to easily improve long-time-horizon spacecraft safety planning through training with an emulation of the spacecraft's implicit RTA (Run Time Assurance) system, while the rule-based policy would take significantly more software development time to reach the same level of safety performance. We also find that the DRL policy is more generalizable than the rule-based policy when increasing complexity of the simulation, since the DRL policy can be easily re-trained/re-tuned. As we increase complexity of C2 requests and number of safety constraints to match realistic scenarios, we expect DRL's optimality, safety, and generalizability benefits will strengthen compared to traditional rule-based and optimization-based policies. This paper also explores approaches for building trust from constellation operators in satellite C2 policies including: prototyping visualizations of expected C2 results in lookahead planning windows, designing methods to allow operators to compare/override policies, and calculating comparative optimality/safety metrics. We also outline next steps to potentially further improve optimality and safety, through: 1) comparing the implemented implicit RTA DRL filter, with an explicit RTA DRL comprehensive training technique and 2) comparing the implemented traditional decentralized satellite onboard control algorithms, with a multi-satellite DRL collaboration algorithm.

INTRODUCTION

There is a growing need for both the DoD (Department of Defense) and commercial satellite operators in reducing satellite C2 request timelines, promoting communications resiliency, and allowing for cost-effective constellation scalability. These needs call for low-SWaP (Size, Weight, and Power) automated satellites with decentralized Command and Control (C2) computation capabilities onboard. In this paper we will focus on C2 of many satellites as a result of pop-up requests from many tactical requestors, while coordinating with one ground station. This concept of operations is provided in Figure 1, along with a list of expected operational benefits for onboard C2 computation. Traditional C2 satellite autonomy policies (i.e. rule-based and optimization-based techniques) are most reliable when using simplistic C2 requests, low-detail representations of satellite sub-systems, and collaborating among few satellites simultaneously [1-6]. However, as we increase complexity in these areas, developing C2 policies using these traditional techniques are computationally expensive, do not generalize well, and often require significant software development budgets. Recent work has demonstrated that Deep Reinforcement Learning (DRL) policies are viable for

* Scientific Systems Company, Inc., 500 W Cummings Park #3000, Woburn, MA 01801
† University of Colorado Boulder, Regent Dr, Boulder, CO 80309
‡ Air Force Research Laboratory, Space Vehicles Directorate, 3550 Aberdeen Ave SE, Kirtland AFB, NM 87117

efficient satellite control, while showing promise for alleviating these issues associated with traditional techniques [7-13]. In this paper, we explore mission scenarios within this concept of operations, while comparing DRL C2 with a traditional rule-based approach for C2.

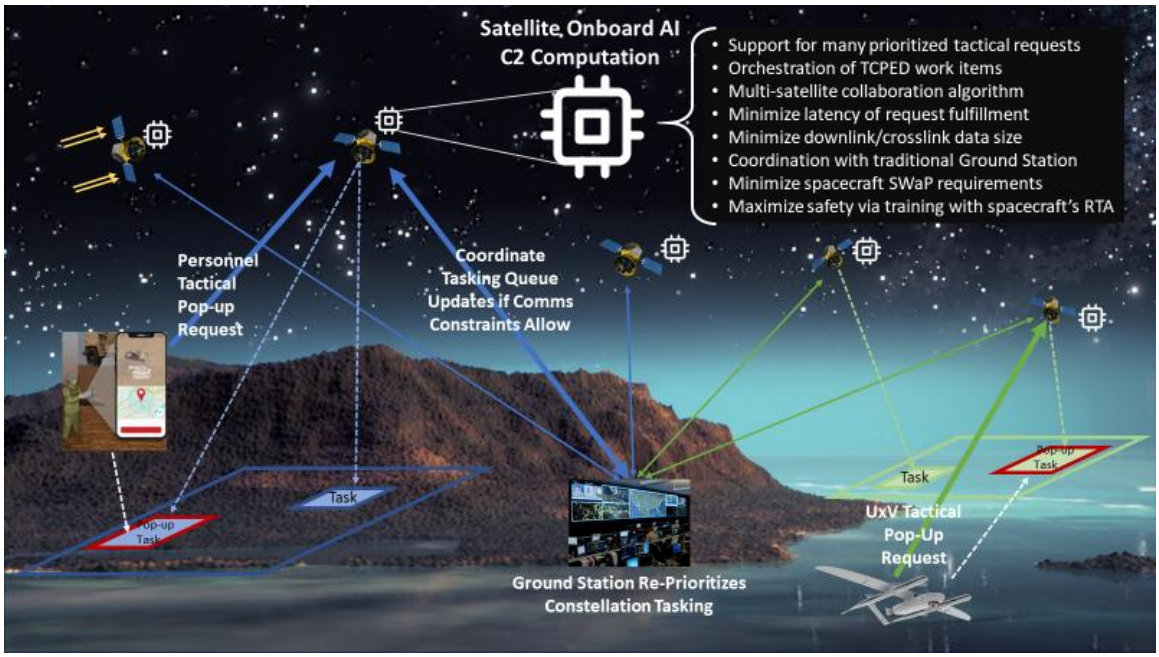

Figure 1: Concept of operations for satellite onboard AI C2 computation

Spacecraft safety is also important to consider when performing onboard C2. Spacecraft manufacturers generally have fail-safe processes, i.e. Run Time Assurance (RTA) algorithms, in place on the spacecraft. RTA methods can range from fully autonomous intervention, to human operator intervention, and are generally decoupled from the main control algorithms to ensure safety. When developing new C2 policies, it's important to consider the relationship with the spacecraft's RTA system, specifically in minimizing the number of times the C2 policy activates RTA intervention. If we avoid all RTA interventions with our C2 policy, we reduce safety risk from a possible RTA intervention malfunction. Minimal RTA interventions also suggests the developed C2 policy has effectively planned long-term spacecraft actions while considering RTA safety constraints. In this paper, we emulate an implicit RTA spacecraft system, and we train the DRL C2 policy within these safety constraints to optimize both safety and long-term planning. Implicit RTA approaches use a predefined procedure to evaluate when to intervene, and explicit RTA approaches define all safe behavior modes the controller is allowed to perform (while often also using a backup implicit RTA approach) [14].

DRL policies for multi-satellite onboard C2 have many benefits, however a significant hurdle is achieving trust from constellation operators. Explainability of DRL's specific algorithmic components is a nascent field of research, but some early results and organizational structures exist to assist the exploration of detailed explainability techniques for satellite C2 DRL [15]. However, it is possible that trust in DRL can be achieved through a set of more generalized explainability methods, which have the significant benefit of allowing for direct comparisons between any satellite C2 autonomy algorithms. As a result, in this paper we focus on general explainability methods applicable to any satellite C2 policies, and also provide example explainability results of both DRL and rule-based algorithms. More specifically, we will explore methods to build trust in any satellite C2 algorithms via: prototyping visualizations of expected C2 results in lookahead planning windows, designing methods to allow operators to compare/override policies, and calculating comparative optimality/safety metrics.

ARCHITECTURE

Our decentralized architecture for C2 is meant to receive requests from any dispersed tactical requestors, and autonomously orchestrate all necessary TCPED (Tasking, Collection, Processing, Exploitation, and Dissemination) work items onboard satellites of the constellation. In order to make a smooth transition to this fully decentralized architecture, we expect that constellation customers will also want full override control from their traditional ground stations. This full control from the ground station includes the abilities of both: reverting to their traditional satellite planning/execution methods, as well as overriding particular policies and actions, described further in the section further below named, "Design of Methods to Allow Operators to Compare/Override Policies." Figure 2 provides a system architecture diagram outlining the functional software components we are considering, while providing a path forward for using existing satellite C2 control systems.

We also emulate the safety RTA system of the spacecraft during the training process, in attempt to never activate the implicit RTA system during DRL policy inference. Including this implicit RTA filter in the training process aims to promote safe spacecraft operation, while optimizing long-term satellite plans.

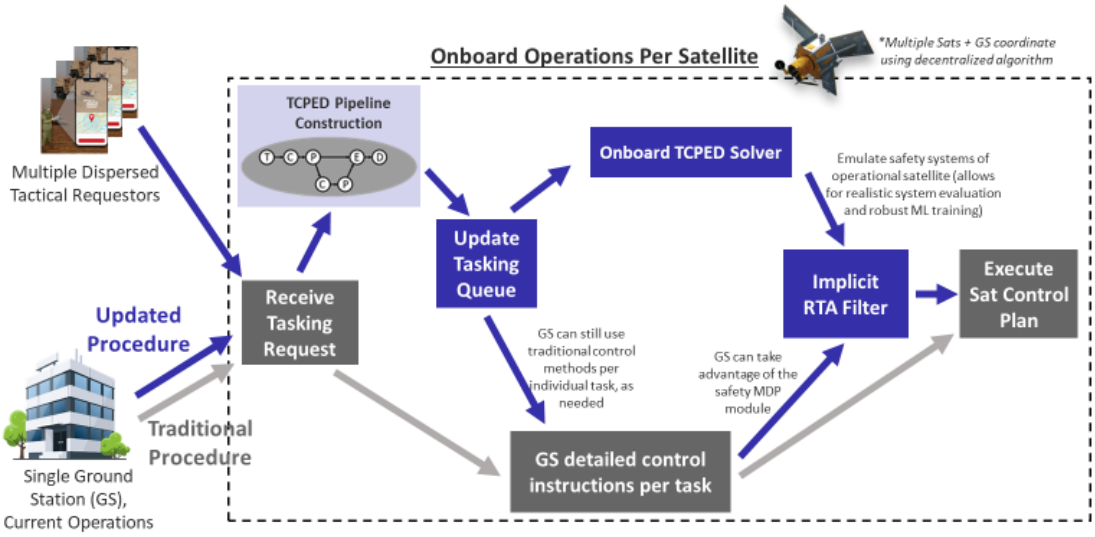

Figure 2: Architecture for demonstrating satellite onboard C2, while integrating with traditional Ground Station operations

SIMULATION ENVIRONMENT

A simulation environment is chosen among the following options for this study: Basilisk[§] [16], NOS3[**], STK[††], OSK[‡‡], FreeFlyer[§§], DSim[***], KubeSat[†††], and AFSIM[‡‡‡]. Criteria for selection included: ability to simulate requests/tasks, fast simulation speed, history of software transition into Hardware-in-the-Loop testing, support of high-fidelity spacecraft constraints (including constraints of attitude, battery, communications, data storage, etc.), ease of development, and support for Machine Learning (ML) algorithm

[§] http://hanspeterschaub.info/basilisk/index.html
[**] http://www.stf1.com/simulation.php
[††] https://www.agi.com/missions/space-operations-missions
[‡‡] https://github.com/OpenSatKit/OpenSatKit
[§§] https://ai-solutions.com/freeflyer-astrodynamic-software/
[***] https://www.psatellite.com/products/simulation-framework/
[†††] https://github.com/IBM/spacetech-kubesat
[‡‡‡] https://www.wpafb.af.mil/News/Art/igphoto/2001709929/

development (via prior integration with ML environments such as OpenAI Gym[§§§]). All criteria were considered for each simulator, and the Basilisk simulation environment was determined to be best suited for this study. Basilisk also has the added bonus of being fully open-source software, allowing for greater potential for community collaboration. Basilisk also has open-source examples for interfacing with OpenAI Gym, allowing for fast deployment and comparisons of ML algorithms. The Basilisk simulation framework's subcomponents are summarized in Figure 3. In our experiments, we reformulate the spacecraft C2 problem as a Partially-Observable Markov Decision Problem (POMDP), amicable to DRL techniques. Both the Basilisk simulation environment and OpenAI Gym wrapper are able to support this problem formulation well.

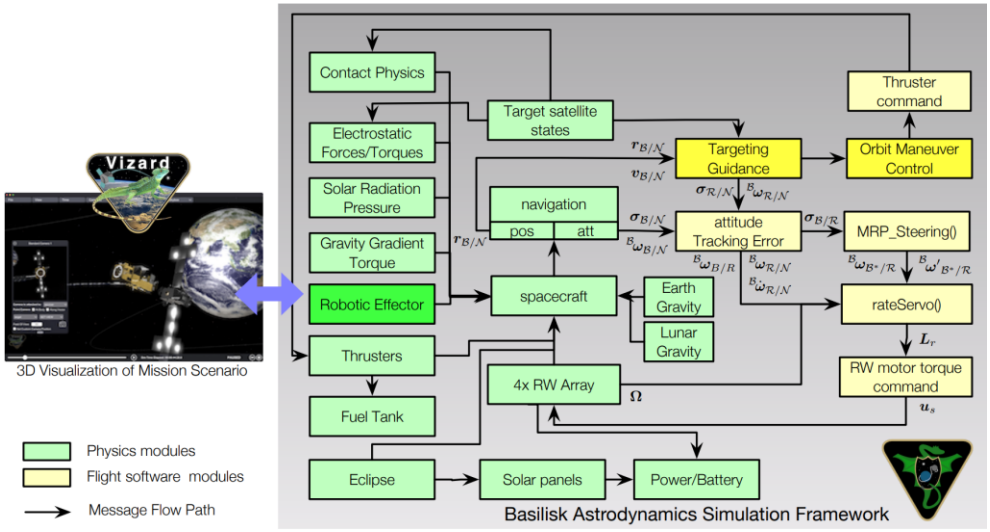

Figure 3: Basilisk simulation environment and Vizard visualization software

COMPARING DRL AND RULE-BASED POLICIES

To compare DRL and rule-based policies, we will first define the constraints that will allow for high-fidelity experimentation, then discuss the respective DRL/rule-based policy algorithmic approaches, and finally discuss results of our experimentation.

Constraints

To perform high-fidelity experimentation, we consider spacecraft safety constraints including: momentum wheel attitude control, momentum wheel dumping actions to avoid spacecraft instability, pointing at the sun for battery recharge, and battery discharge as a result of momentum wheel actions and imaging. To add operational realism, we also consider hundreds of prioritized imaging requests (high/medium/low priority) that also have the options of being canceled and overridden. Once the simulation starts, we have the choice to perform one of three actions every few minutes with any C2 policy: 1) point at the sun to recharge, 2) dump momentum from momentum wheels, or 3) point at a target and attempt to take an image. For this paper, we set up the Basilisk simulator with the constraints and parameters listed in Table 1. In these experiments, we assume perfect communication among all satellites, requestors, and ground stations. We also do not consider data storage or communication downlink/crosslink constraints. Figure 4 also provides an example mission where we track some of these safety constraints over time, as a test case for evaluating if our simulation setup is behaving appropriately.

[§§§] https://github.com/Farama-Foundation/Gymnasium

Table 1: Basilisk simulation constraints and parameters

High-Level Parameters	Value	Units
Number of satellites	10	-
Number of targets	1000	-
Number of Requestors	12	-
Number of Ground Stations	1	-
Simulation length	810	min
Step duration	3	min
Percentage of requests that are canceled/replaced	10	%
Orbital elevation	500	km
Orbital trajectory	random	-
Maximum imaging range	5000	km
Minimum elevation for imaging	30	deg
Prioritized targets lookahead window	10	min
Maximum imaged targets per lookahead window	5	-
Number of momentum wheels	3	-
Maximum momentum wheel speed	3000	rpm
Maximum battery power available	180	kJ
Minimum battery power	0	kJ

Low-Level Parameters	Value	Units
Spacecraft mass	300	kg
Spacecraft width	1.38	m
Spacecraft depth	1.04	m
Spacecraft height	1.5	m
Atmospheric density	1.22	kg/m^3
Atmospheric scale	8e3	m
Initial attitude angles	random	rad
Initial attitude rates	random	rad/s
Initial momentum wheel speeds	random	rad/s
Disturbance torque magnitude	2e-4	Nm
Disturbance torque direction	random	rad
Sim dynamics step size	0.5	s
Sim Flight Software step size	1	s
Maximum spin rate	0.001	rad/s
Momentum wheel dumping rate	4	Nms
Continuous power draw	5	W
Solar panel area	0.09	m^2
Solar panel efficiency	20	%

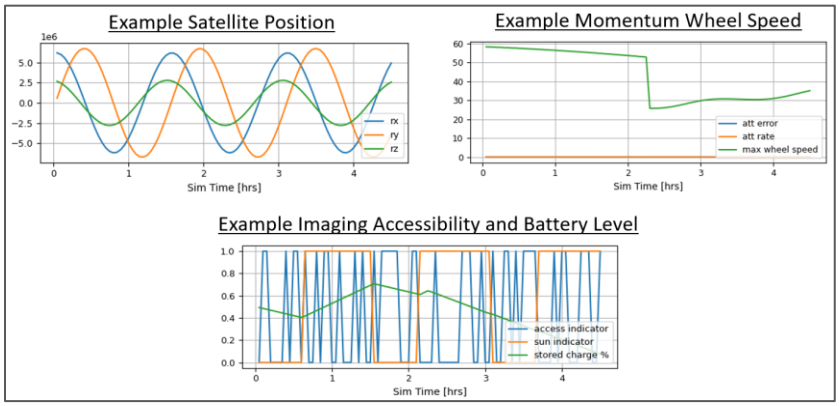

Figure 4: Example mission with measurements of satellite position, momentum wheel speed, imaging accessibility, and battery level percentage

DRL/Rule-Based Policy Algorithms

The DRL policy is trained with a Monte Carlo Tree Search technique on a POMDP representation of this C2 problem, using the same methods as outlined in [13]. The spacecraft's implicit RTA system is emulated using a "correct-by-construction shield synthesis" approach, adapted for the satellite C2 domain to be known as a "discrete safety MDP (Markov Decision Process)" as described in [11]. This Safety MDP exists in parallel with the continuous POMDP representation of the C2 problem space. The DRL training process aims to maximize the number of imaged targets, while avoiding activations of the Safety MDP. On the other hand, the rule-based policy uses a "greedy" approach, where it always tries to attempt to image targets at the next possible opportunities, while also including minor spacecraft safety consideration logic for avoiding unsafe battery and momentum wheel behavior. Both DRL and rule-based policies use the Safety MDP filter after control policy actions have been chosen, in order to emulate spacecraft implicit RTA intervention behavior.

For our experiments, we've designed both DRL and rule-based policies to use the same multi-satellite collaboration algorithm, using a "greedy decentralized" approach. In this "greedy decentralized" multi-satellite collaboration algorithm, for each candidate target, we assign the satellite whose trajectory (during the full simulation lookahead period) is closest to this candidate target.

To consider target priorities (high/medium/low) within our experiments, both DRL and rule-based policies also use the same "priority filtering algorithm." This "priority filtering algorithm" assumes a maximum number of targets that can be imaged within a short lookahead window, where these values are derived from average behavior of our simulator parameters. This "priority filtering algorithm" determines if too many targets are in the lookahead window, and drops the lowest priority targets exhaustively until considering to drop the next-lowest priority targets (i.e. drop low priority targets one-by-one until all are dropped, followed by medium priority targets, followed by high priority targets).

Results of DRL/Rule-Based Policy Comparisons

Initial verification of the DRL/rule-based policies was conducted via compilation of select scenarios into Basilisk's visualization tool, Vizard [16]. Figure 5 provides an example image of a DRL policy during these simulations. The following are more complete video examples of this testing, with references to the full video simulations:

- De-tumbling using momentum wheels, sun pointing for recharge, and momentum dumping[****]
- Nadir pointing while in the reach cone of target imaging location[††††]
- Two satellites going into nadir pointing and recharging[‡‡‡‡]
- Satellite able to change imaging locations, showing DRL model is generalizable[§§§§]

Figure 5: Example DRL test where spacecraft de-tumbles, solar panel sun-pointing maneuver occurs for recharging batteries, and reaction wheel momentum dumping occurs using thrusters

Next, we run the simulation with the exact parameters of Table 1, this time including the full amount of 10 satellites and 1000 targets. The parameters in Table 1 with values labeled "random," are randomly generated each simulation run, for a total of 100 runs. In each of these 100 runs, we compare the DRL and rule-based policies on the same set of randomized initial conditions, and provide the results in Table 2. DRL and rule-based action policy trials use the same "greedy decentralized" multi-satellite collaboration algorithm and the same prioritization filter, resulting in the shared results between DRL/rule-based action policies for those respective entries in the bottom half of the table.

[****] https://www.youtube.com/watch?v=05uiQRZTXoE&t=12s
[††††] https://www.youtube.com/watch?v=nKEUmxyoF5E&t=7s
[‡‡‡‡] https://www.youtube.com/watch?v=0GN7V0rG20w
[§§§§] https://www.youtube.com/watch?v=QUa8mTsOjfk&t=2s

Table 2: Simulation results for both rule-based and DRL policies

Measurement Per Each of the 10 Satellites per Run	Aggregation Across 100 Runs	Rule-Based Policy	DRL Policy
Percentage of Safety MDP activations, across all action steps	Average	0.0685	0
	Std. Dev.	0.1524	0
Computation time for the action policy, including Safety MDP processing	Average	0.20 seconds	0.93 seconds
	Std. Dev.	0.004 seconds	0.031 seconds
Total targets imaged	Average	33.77 targets	34.09 targets
	Std. Dev.	5.34 targets	5.26 targets
Total targets assigned after "greedy" multi-satellite collaboration algorithm	Average	100.0 targets	
	Std. Dev.	36.43 targets	
Total targets able to be imaged based on imaging constraints	Average	45.87 targets	
	Std. Dev.	10.11 targets	
Total targets after prioritization filter	Average	40.19 targets	
	Std. Dev.	6.81 targets	
Computation time for "greedy" multi-satellite collaboration algorithm	Average	4.93 seconds	
	Std. Dev.	1.69 seconds	
Computation time for remaining simulation	Average	9.66 seconds	
	Std. Dev.	4.95 seconds	

Table 2 results show that the DRL policy is able to effectively avoid unsafe behavior via a 0% rate of Safety MDP activations, while the rule-based policy has a high rate of Safety MDP activations for 6.85% of actions taken. The rule-based policy thus introduces an avenue for added safety risk, while the DRL policy does not. We also notice a longer computation time for the DRL action policy than the rule-based policy, likely due to the DRL's neural network inference processing, compared to the rule-based policy's simple logic. We notice the DRL policy has a slightly higher number of targets imaged compared to the rule-based policy.

We also notice significant benefits to software development time and generalizability for the DRL policy compared to the rule-based policy. Once the simulation environment is created, we find the DRL policy is able to easily improve long-time-horizon spacecraft safety planning through training with an emulation of the spacecraft's implicit RTA system, while the rule-based policy would take significantly more software development time to reach the same level of safety performance. We also find that the DRL policy is more generalizable than the rule-based policy when increasing complexity of the simulation, since the DRL policy was able to be easily re-trained/re-tuned on new simulation constraints through our software development process.

As we increase complexity of C2 requests and number of safety constraints to match realistic scenarios, we expect the benefits of DRL's optimality, safety, generalizability, and software development time will strengthen compared to traditional rule-based and optimization-based policies. To match more realistic scenarios and to explore these DRL benefits further, future research can focus on:

- Adding additional mission capabilities (e.g. optimize data storage / crosslinks / downlinks, add ability to specify/optimize image pixels-on-target, etc.)
- Increasing number of safety constraints (e.g. sensor thermal constraints, trajectory constraints, etc.)
- Incorporating more complex tasking (task dependencies, larger variety of TCPED work items, etc.)
- Incorporating a multi-satellite collaboration DRL technique (discussed further below)

BUILDING TRUST IN AI FOR SATELLITE C2

A large hurdle for AI/DRL satellite C2 policies to reach mass adoption is the issue of achieving trust from constellation operators. Here we will discuss methods of building trust via: prototyping visualizations of expected C2 results in lookahead planning windows, designing methods to allow operators to compare/override policies, and calculating comparative optimality/safety metrics for any C2 policy.

Prototyping Visualizations of Expected C2 Results in Lookahead Planning Windows

A basic step towards AI policy explainability is providing visualizations and relevant statistics to satellite operators, and evaluating the policy's alignment with their needs. An example visualization method is provided in Figure 6, used during our experimentation. This method includes features we believe are important to satellite operators, including the abilities to:

- Input a "lookahead window" time period (e.g. 24 hours), run the expected onboard DRL policies for this time period, and use a "time scale slider" interaction to analyze expected future behaviors
- View all future projected satellite trajectories
- View all requests categorized by their stage in the policy's process, and their priorities (P1, P2, P3)
- Status of requestors and ground station, including their total targets imaged, pending, and canceled
- View the average Safety MDP activation percentage over time, helpful to ensure we are not activating the spacecraft's emulated RTA with the given policy

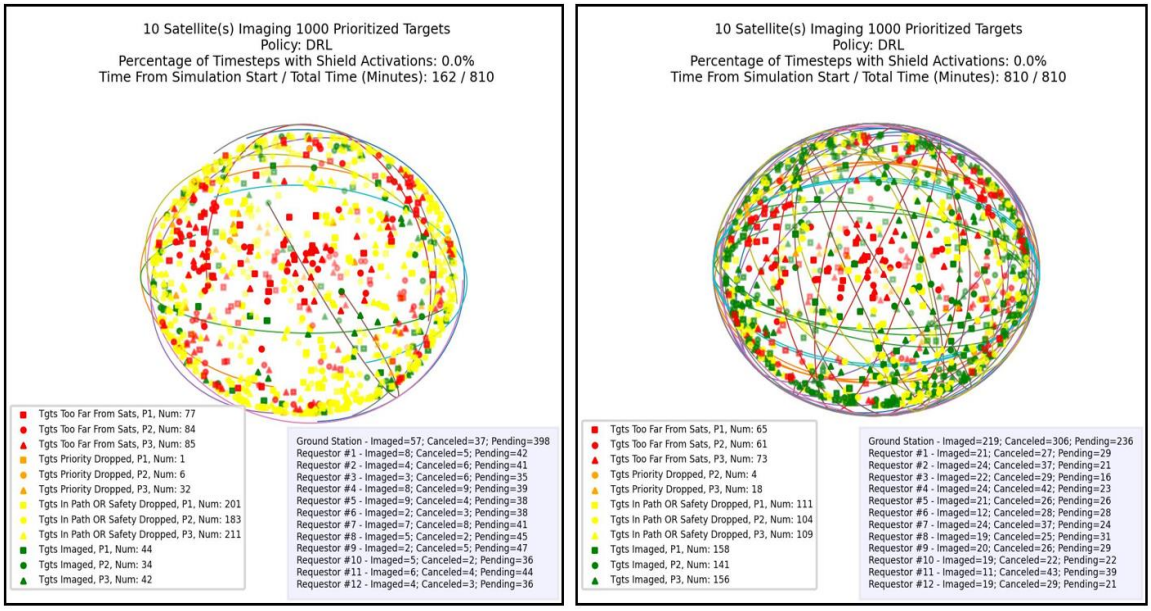

Figure 6: Constellation C2 policy visualization method, at the beginning (*left*) and end (*right*) of the simulation period

Designing Methods for Allowing Operators to Compare/Override Policies

As we transition to using AI C2 methods, we want to allow constellation operators to continue use of their previous spacecraft policy methods, as well as being flexible in using the latest developed policy algorithms. We have also noticed that operators also sometimes want full control over the C2 process, to a level as detailed as assigning one particular satellite to one particular task. As a result, we propose an editable policy structure, where at any future timestep of a chosen policy, the operator is able to change: any specific satellite mission/safety actions, any satellite/request pairing, as well as being able to change the policy type.

We also expect that for one set of initial conditions, operators will want to audit/compare various policies (including any additional policy override features), to help them understand differences in performance as we propagate these actions forward in time. However, it is expected that as we work with problem spaces of higher dimensionalities (e.g. more satellites/requests, etc.), and higher complexities (e.g. more safety/mission constraints, etc.), initial guesses from operators on policy overrides will be suboptimal compared to trained DRL policies. Figure 7 provides an example comparison between a DRL and rule-based policy, where an operator can compare visualizations and summary statistics over time. In this example we can see that the DRL policy has a rate of 0% Safety Filter "Shield" activations, while the rule-based policy has a rate of 7.4%.

We also notice three more targets being imaged using the DRL policy compared to the rule-based policy. It is difficult to see these imaged target differences in the visualizations, so a future upgrade can include the highlighting of differences between these comparative visualizations/statistics. These override and comparison methods are expected to be useful in constellation pre-mission planning, potential interventions during missions, and post-mission debrief analysis.

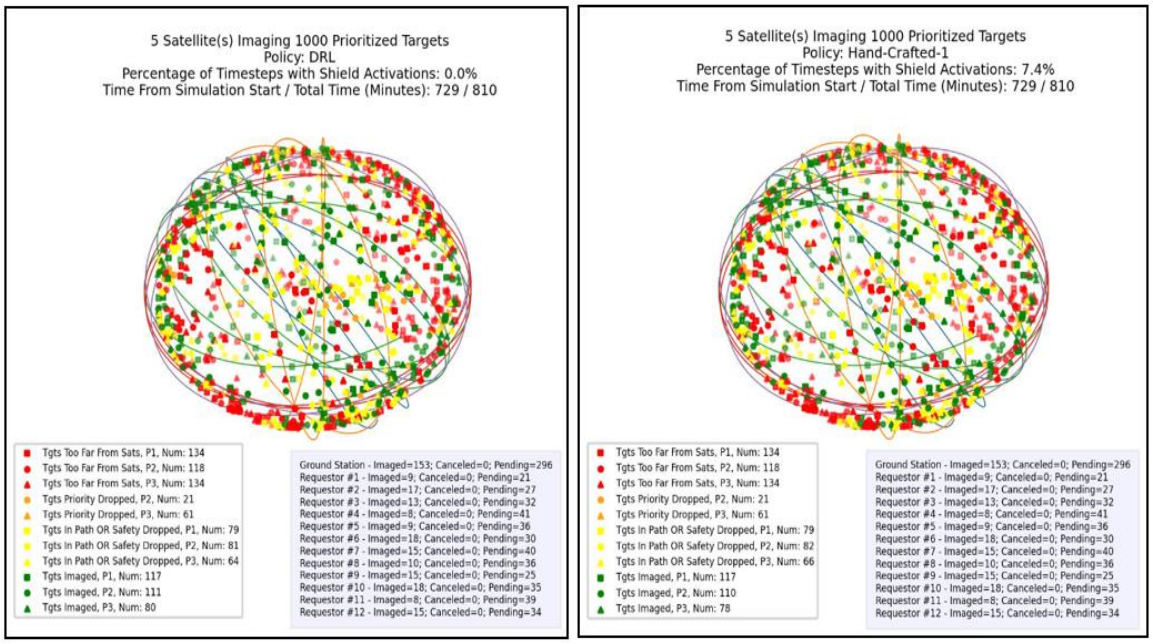

Figure 7: Example comparison between a DRL policy (*left*) and rule-based policy (*right*)

Calculating Comparative Optimality/Safety Metrics for C2 Planning Windows

It can also be helpful to constellation operators to provide fast verification methods compatible with any policies, to quantify their trust in upcoming satellite control decisions. During simulated lookahead planning windows, operators may often not have time to understand the level of detail provided in Figure 6 and Figure 7. Instead they may want trustworthy summary statistics on how the proposed policies will behave in a certain lookahead window. Below are two core metrics that will help operators judge the optimality and safety of any given satellite C2 policy:

- **Optimality metric:** Calculated by using a "theoretically optimal" approach to determine how many requests can possibly be fulfilled given a set of initial conditions, then dividing the candidate policy's projected completed requests by this result. "Theoretically optimal" request fulfillment is calculated by running a simulation during a given time period using the following algorithms, and measuring the number of projected completed requests:
    - Spacecraft safety constraints, prioritization filter, and spacecraft dynamics: we ignore all these algorithmic steps, in order to avoid bias resulting from sequences of events which won't be the exactly the same among different policies we would like to compare using this metric.
    - Action policy: "greedy action policy" where we attempt to image/collect data at each possible chance.
    - Multi-satellite collaboration algorithm: the "greedy decentralized" algorithm will provide the theoretically optimal behavior: for each candidate target, we assign the satellite whose trajectory (during the full simulation lookahead period) is closest to this candidate target.
- **Safety metric:** Calculated as the percentage of timesteps the Safety MDP filter is activated, effectively counting how often the spacecraft's implicit RTA system needs to intervene. This

"Percentage of Safety MDP activations" is subtracted from 100% to calculate the final Safety metric, where a higher Safety metric means we have safer expected behavior.

We extend the experimental results of Table 2, to calculate these Optimality and Safety metrics. To calculate the Optimality metric, the "theoretical optimal performance" precursor is equivalent to Table 2's entry "Total targets able to be imaged based on imaging constraints," namely 45.87 targets per satellite on average, across the 100 experimentation runs. With the Table 2's "Total targets imaged" values for the rule-based and DRL policies of 33.77 average targets and 34.09 average targets respectively, dividing by the "theoretical optimal" of 45.87 average targets will give us their Optimality metrics. The final Optimality metrics for the rule-based and DRL policies are thus 73.6% and 74.3% respectively. The Safety metrics for the rule-based and DRL policies are equivalent to 100% minus Table 2's "Percentage of Safety MDP activations" (6.85% and 0.00% respectively), resulting in Safety metrics for the rule-based and DRL policies of 93.2% and 100% respectively. Applicable data and final results for the Optimality and Safety metrics are provided in Table 3.

Table 3: Applicable data and calculated Optimality and Safety metrics

Measurement Per Each of the 10 Satellites, Averaged Across 100 Runs	Rule-Based Policy	DRL Policy
Total targets able to be imaged based on imaging constraints, "theoretical optimal"	45.87 targets	
Total targets imaged	33.77 targets	34.09 targets
Optimality Metric	73.6%	74.3%
Percentage of Safety MDP activations, across all action steps	6.9%	0.0%
Safety Metric	93.2%	100.0%

These Optimality and Safety metric calculation methods provide an opportunity for universal comparisons among any satellite C2 policy's resulting plan (e.g. traditional ground station's plan, a DRL policy's plan, etc.) to help operators quickly verify they have trust in the upcoming satellite control decisions.

FUTURE WORK

Improving C2 Efficiency via Explicit RTA DRL Training on Safety Constraints

In this paper we study an implicit RTA approach to support safe DRL training and accurate emulation of spacecraft C2 execution. There is an opportunity to improve C2 efficiency by replacing this implicit RTA approach with the training an explicit RTA DRL policy, while also including a more flexible implicit RTA filter as a last fail-safe step. The explicit RTA system actions (including momentum wheel commands, thruster commands, etc.) can be trained alongside mission actions (including pointing at targets, downlinking data, etc.) within a single DRL policy, to allow for a full action search space during training of the DRL policy, resulting in more optimal safety/mission performance compared to the initial implicit RTA approach (which has no consideration of mission action optimization). Techniques also exist to further enhance safety of explicit RTA DRL policies via Neural Network (NN) verification techniques [17]. However, developing explicit RTA DRL policies is a longer-term objective, because it requires significant coordination with the spacecraft manufacturing process, for accessing/testing all safety control bus commands. Note that safety is unlikely to be fully guaranteed through an explicit RTA DRL policy, thus we still recommend a final, fail-safe, implicit RTA step following all DRL policy action recommendations. Design of this final implicit RTA step can also be informed by the explicit RTA DRL policy's behavior, via DRL explainability techniques [15].

By using an explicit RTA DRL training approach that has access to all spacecraft actions, we also have an opportunity to recognize and correct spacecraft subsystem degradation. To accomplish this, the NN input layer can consider detailed sensor data and a history of all past spacecraft actions, which enables NN training for monitoring and adjusting future spacecraft actions. As an example, with these extra detailed sensor inputs and historical action inputs, the NN could automatically detect momentum wheel degradation, and plan for additional reorientation time, thus promoting safer operation and more precise long-term planning.

Training of an explicit RTA DRL policy can be optimized for multiple safety constraints at once (including battery charge, momentum wheel speeds, spacecraft sensor heat constraints, etc.), by maximizing a "Safety Evaluation Metric" in the DRL training process, among other mission-based evaluation metrics. We can follow the "reliability of a series system" calculation method [18], by multiplying all individual safety buffer "reliabilities" together, to provide an overall Safety Evaluation Metric. This "reliability of a series system" formulation is important because we want all individual safety "reliability" metrics to collectively have large safety buffers from their operational safety limits, while also ensuring that the overall Safety Evaluation Metric scores poorly when even just one safety constraint limit is approached too closely. For example, the safety buffer "reliability" of a battery charge at 0% will be 0%; and for higher battery charge values, we can exponentially increase the battery charge "reliability" score to 100% (e.g. a score of 100% is achieved when dynamic conditions are met where there is minimal risk of full battery discharge to occur, such as when there is a long time until we reach the next period of earth's shadow). We expect that training of explicit RTA DRL using this comprehensive Safety Evaluation Metric will maximize all safety buffers in unison (i.e. maximize the gaps between safety constraint limits and their operational levels), while also having the ability to optimize request fulfillment performance with additional weighted model evaluation criteria.

AI Technique for Multi-Satellite Collaboration

Implementations do not yet exist for multi-satellite collaborative C2 using a purely Machine Learning approach. However, the research results of Multi-Agent Reinforcement Learning (MARL) algorithms is a promising area for this domain. Single-satellite DRL trained in an OpenAI Gym[*****] environment is extendable to MARL training using compatible simulation frameworks such as PettingZoo[†††††], providing a lower barrier to entry when extending the open-source Basilisk OpenAI Gym environments. The multi-satellite collaboration ability that MARL would provide competes with traditional multi-satellite collaboration algorithms (greedy, consensus, auction/bid, etc.), thus it's recommended to evaluate the following criteria between traditional multi-satellite collaboration C2 algorithms and MARL C2: constellation scalability, request fulfillment optimality, communications bandwidth requirements, and request fulfillment speed.

CONCLUSION

We studied multi-satellite decentralized onboard C2, comparing a Deep Reinforcement Learning (DRL) control policy with a rule-based policy. We considered hundreds of dynamic prioritized imaging requests from dispersed tactical requestors, routed to many satellites, and with periodic ground station coordination. We simulated spacecraft actions including: momentum wheel attitude control, momentum wheel dumping actions, pointing at the sun for battery recharge, and battery discharge as a result of momentum wheel actions and imaging. Compared to a rule-based policy, the DRL policy showed slightly better request fulfillment performance and slightly worse computation time. Once the simulation environment is created, we found the DRL policy is able to easily improve long-time-horizon spacecraft safety planning through training with an emulation of the spacecraft's implicit RTA (Run Time Assurance) system, while the rule-based policy would take significantly more software development time to reach the same level of safety performance. We also found that the DRL policy is more generalizable than the rule-based policy when increasing complexity of the simulation, since the DRL policy can be easily re-trained/re-tuned. We also explored approaches for building trust from constellation operators in satellite C2 policies including: prototyping visualizations of

[*****] https://github.com/Farama-Foundation/Gymnasium
[†††††] https://github.com/Farama-Foundation/PettingZoo

expected C2 results in lookahead planning windows, designing methods to allow operators to compare/override policies, and calculating comparative optimality/safety metrics. We also outlined next steps to potentially further improve optimality and safety, through: 1) comparing the implemented implicit RTA DRL filter, with an explicit RTA DRL comprehensive training technique and 2) comparing the implemented traditional decentralized satellite onboard control algorithms, with a multi-satellite DRL collaboration algorithm.

ACKNOWLEDGEMENTS

This publication is based upon work sponsored by Air Force Research Laboratory (AFRL) under AFRL Contract FA864922P0833, with AFRL Public Approvals Case Number AFRL-2023-0173. Any opinions, findings, conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of AFRL, the U.S. Air Force, the U.S. Department of Defense, or the U.S. Government.

REFERENCES

[1] Kubitschek, D. G., "Impactor Spacecraft Encounter Sequence Design for the Deep Impact Mission," Jet Propulsion, 2005, pp. 1–14. URL https://smartech.gatech.edu/handle/1853/8031.

[2] Foster, C., Mason, J., Vittaldev, V., Leung, L., Beukelaers, V., Stepan, L., and Zimmerman, R., "Constellation Phasing with Differential Drag on Planet Labs Satellites," Journal of Spacecraft and Rockets, Vol. 55, No. 2, 2018. doi:10.2514/1.A33927, URL, https://arc.aiaa.org/doi/pdf/10.2514/1.A33927.

[3] Chien, S., Sherwood, R., Tran, D., Castano, R., Cichy, B., Davies, A., Rabideau, G., Tang, N., Burl, M., Mandl, D., Frye, S., Hengemihle, J., Agostino, J. D., Bote, R., Trout, B., Shulman, S., Ungar, S., Gaasbeck, J. V., Boyer, D., Systems, C., Griffin, M, , and Mit, H.-h. B., "Autonomous Science on the EO-1 Mission," Proceedings of International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), No. May, 2003.

[4] Chien, S., and Jonsson, A., "Automated Planning & Scheduling for Space Mission Operations JPL," , No. February, 2005.

[5] Chien, S. A., Tran, D., Rabideau, G., Schaffer, S. R., Mandl, D., and Frye, S., "Timeline-Based Space Operations Scheduling with External Constraints," Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS), No, Icaps, 2010, pp. 34–41.

[6] Choo, T. H., and Skura, J. P., "SciBox: A software library for rapid development of science operation simulation, planning, and command tools," Johns Hopkins APL Technical Digest (Applied Physics Laboratory), Vol. 25, No. 2, 2004, pp. 154–161.

[7] B. Gaudet, R. Linares, and R. Furfaro, "Adaptive Guidance and Integrated Navigation with Reinforcement Meta-learning," Acta Astronautica, Vol. 169, April 2020, pp. 180–190, 10.1016/j.actaastro.2020.01.007.

[8] D. M. Chan and A. Agha-mohammadi, "Autonomous Imaging and Mapping of Small Bod ies Using Deep Reinforcement Learning," 2019 IEEE Aerospace Conference, 2019, pp. 1–12, 10.1109/AERO.2019.8742147.

[9] A. Harris, T. Teil, and H. Schaub, "Spacecraft Decision-Making Autonomy Using Deep Reinforcement Learning," AAS Spaceflight Mechanics Meeting, Maui, Hawaii, January 13–17 2019. Paper No. AAS 19-447.

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

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

[12] A. Herrmann and H. Schaub, “Monte Carlo Tree Search with Value Networks for Autonomous Space craft Operations,” AAS/AIAA Astrodynamics Specialist Conference, Lake Tahoe, CA, Aug. 9-13 2020. AAS 20-473.

[13] A. Herrmann and H. Schaub, “Autonomous Spacecraft Tasking using Monte Carlo Tree Search Methods,” AAS/AIAA Space Flight Mechanics Meeting, Charlotte, NC, January 31–February 4, 2021.

[14] T. Gurriet, M. Mote, A. Singletary, E. Feron, and A. D. Ames, “A scalable controlled set invariance framework with practical safety guarantees,” in 2019 IEEE 58th Conference on Decision and Control (CDC). IEEE, 2019, pp. 2046–2053.

[15] S. Milani, N. Topin, M. Veloso and F. Fang, "A Survey of Explainable Reinforcement Learning," 2022, arXiv:2202.08434v1.

[16] Kenneally, P. W., Piggott, S., and Schaub, H., “Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework,” Journal of Aerospace Information Systems, Vol. 17, No. 9, Sept. 2020, pp. 496–507, doi:10.2514/1.I010762.

[17] M. Fazlyab, M. Morari and G. J. Pappas, "Safety Verification and Robustness Analysis of Neural Networks via Quadratic Constraints and Semidefinite Programming," in IEEE Transactions on Automatic Control, vol. 67, no: 1, pp. 1-15, Jan. 2022, doi: 10.1109/TAC.2020.3046193.

[18] Menčík, Jaroslav. "Reliability of Systems". Concise Reliability for Engineers, edited by Jaroslav Mencik, IntechOpen, 2016. 10.5772/62358.