Domain-Free and Domain-Informed Reinforcement Learning for the Agile Earth-Observing Satellite Target Sequencing Problem

Mark Stephenson

***Abstract*—The agile Earth-observing satellite scheduling problem aims to maximize observation profit while managing resources. A major subproblem is the sequencing problem, which seeks to find a method of identifying the next target to image that optimizes the total reward for heterogeneously-valued and distributed targets. By formulating the sequencing problem as an Markov decision problem (MDP), reinforcement learning (RL) techniques may be applied to select optimal targets on the fly. In this paper, two RL methods are tested: a dueling deep Q-network (DQN), which learns a policy without prior knowledge, and advantage error learning, which attempts to improve upon a know suboptimal advantage heuristic. While dueling DQN demonstrates that RL is feasible on this problem, neither method was able to exceed the performance of the heuristic.**

I. Introduction

The agile Earth-observing satellite (AEOS) scheduling problem aims to maximize the number and value of ground targets imaged by an orbiting satellite while managing resources [1]. “Agile” indicates that the satellites can not only slew side-to-side to capture off-track targets but also slew forward and backward along-track. The transition time between two targets is both target and time dependent. Particularly in environments with a high density of targets, finding an optimal target sequences is both challenging and rewarding.

Recently, reinforcement learning (RL)-based approaches to the AEOS scheduling problem (AEOSSP) have been considered. These are opposed to more traditional methods that preplan activity sequences using mixed-integer programming [2]–[4] or other methods [5]–[7]. However, policy-based solutions are desirable because they are non-brittle and adaptable in changing environments. Wei et al. [8] use deep RL to adapt solutions to the conventional (rolling slews only) EOS problem for use in the AEOSSP. Piccinin et al. [9] use deep Q-learning to find a policy for a similar problem, the autonomous mapping of small bodies. Herrmann et al. [10], [11] use PPO and MCTS-learn to solve the AEOSSP with safety constraints under the assumption of fixed decision intervals.

In this paper, the target sequencing subproblem of the AEOSSP is approached using two RL methods. The first, a dueling deep Q-network (DQN), learns without any domain knowledge. The second, advantage error learning, attempts to learn an improved policy on top of domain knowledge from an initial heuristic policy. The latter is a novel technique inspired by other error-learning networks [12].

PhD Student, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, Boulder, CO, 80303. AIAA Member.

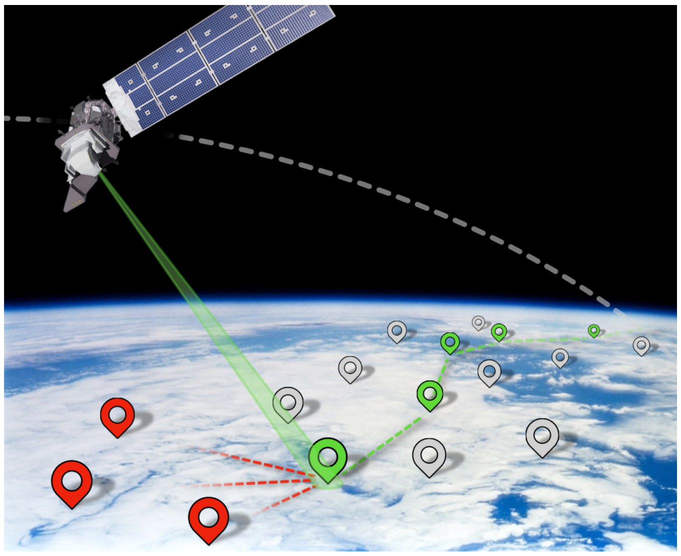

Fig. 1: The imaging satellite must select which of the upcoming (red) targets to attempt to image next.

II. Problem Statement

A. Agile Earth-Observing Satellite Scheduling Problem Formulation

The AEOSSP is considered for a single satellite over a three-orbit horizon (this horizon typically would be infinite and across arbitrary orbits, but to limit scope for this project a finite horizon on a fixed orbit is considered). The satellite is on an orbit with prescribed position trajectory $\boldsymbol{r}(t)$ and has some attitude $\boldsymbol{\sigma}$ and body rate $\boldsymbol{\omega}$, both of which depend on prior controls. The satellite also has an imaging instrument pointing in direction $\hat{\boldsymbol{c}}$ which is fixed relative to the body at ${}^{\mathcal{B}}\hat{\boldsymbol{c}}$. Let $\boldsymbol{x} = \{\boldsymbol{r}, \dot{\boldsymbol{r}}, \hat{\boldsymbol{c}}, \boldsymbol{\omega}\}$ be the satellite’s dynamic state; the instrument pointing direction is used rather than attitude, as it is assumed that the spacecraft operates equivalently up to rotations about the boresight. The satellite’s components and software are modelled fully and realistically using the Basilisk [13] astrodynamics framework; however, resource constrains are not considered in this work by equipping the satellite with generous data and power storage capacities.

The environment consists of targets $c \in T$. A target c_i is a tuple $(\boldsymbol{r}_i, \rho_i)$ consisting of its position $\boldsymbol{r}_i$ and value ρ_i. The set $I \subset T$ consists of imaged targets. A target is imaged (added to I) when the spacecraft’s imaging instrument $\hat{\boldsymbol{c}}$ is pointing at the target within an angle threshold $\delta\theta$ and rate threshold $\delta\omega$

and the spacecraft is within the target's view cone. The view cone is defined by the target zenith vector and a half-angle ϕ, the compliment of the satellites elevation angle relative to the target, satisfying

$$\angle\left(\boldsymbol{r}_i, \boldsymbol{r}(t) - \boldsymbol{r}_i\right) \leq \phi \tag{1}$$

The imaging window for a target is $t \in [\tau_i^o, \tau_i^c]$, during which Equation 1 is satisfied for target u_i. Since $\boldsymbol{r}(t)$ is known, a numerical root finding method can quickly precompute all imaging windows.

The ordered set U consists of unimaged targets $u \in U \subset T$ ordered by the close time τ^c of the current or soonest imaging opportunity. At some time t_u, the set U is defined as

$$U(t_u) = \{u_i \mid u_i \in T \setminus I \text{ if } \tau_i^c > t_u\} \quad \text{ordered by} \quad \tau_{i+1}^c \geq \tau_i^c \tag{2}$$

The goal of the AEOS target sequencing problem is to maximize the total reward of imaged targets:

$$\text{maximize} \sum_{c_i \in I} \rho_i \tag{3}$$

In practice, this translates into two problems that a scheduler must solve:

1) Is there sufficient time to transition and image a target before it goes out of range?
2) Is it more advantageous to image a low-value target quickly or take a longer time to image a higher value target.

Together, these factors yield a travelling salesman-like problem with uncertainty in the existence of edges between vertices.

B. Markov Decision Process

The AEOSSP is modeled as a Markov decision process (MDP) [14] with tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$: state space, action space, transition function, and reward function.

The state space $\mathcal{S}$ is expressed as a the concatenation of the current satellite and environment state spaces, where the state $s = \{\boldsymbol{x}, U_N\}$. The satellite's dynamic state $\boldsymbol{x} = \{\boldsymbol{r}, \dot{\boldsymbol{r}}, \hat{\boldsymbol{c}}, \boldsymbol{\omega}\}$, as defined above. The environment's state U_M is the list of M upcoming targets in the form $\{(\boldsymbol{r}_i, \rho_i) \mid i \in [1, M]\}$. In simulation, the system only approximately adheres to the Markov assumption as other hidden states exists that evolve with and influence the model, such as roll angle about the boresight, battery charge, and reaction wheel speeds. Also, targets beyond U_M are not known to the agent; this serves to limit the state vector to a reasonable length. In practice, these factors can be taken as a source of uncertainty in the transition dynamics. While this could be modelled as a partially-observable MDP (POMDP), that formulation complicates the problem without providing useful insight.

In this sequencing-only problem, the action space $\mathcal{A}$ consists of attempting to image a target in U. Only a finite subset of the next $N < M$ targets are considered as reasonable actions. Action a_i corresponds to imaging target $u_i \in U$. When a_i is selected, a pointing control algorithm on the spacecraft is activated that guides the instrument $\hat{\boldsymbol{c}}$ to align with $\boldsymbol{r}_i - \boldsymbol{r}(t)$, the spacecraft-to-target vector.

The state transition probability function $\mathcal{T}$ is deterministic (when accounting for hidden states in the simulation environment that are not directly exposed in the state vector), so a generative model for state transitions $\mathcal{G} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is used which propagates the satellite's dynamics under control of the targeting mode associated with action a_i. The system is propagated until one of two conditions is met:

1) The currently selected target u_i is successfully imaged: $u_i \in I$
2) The currently selected target's imaging window closes before being imaged: $t > \tau_i^c \wedge u_i \notin I$

The reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ assigns reward for successfully imaging the selected target by checking if the target associated with action a_i has been added to the imaged set I during the state transition:

$$\mathcal{R}(s, a_i, s') = \begin{cases} \rho_i & \text{if } u_i \notin I_s \text{ and } u_i \in I_{s'} \\ 0 & \text{else} \end{cases} \tag{4}$$

The policy $\pi(s)$ selects an action in $\mathcal{A}$ based on the current state s. The objective of this work is to find an optimal policy that maximizes the sum of rewards (equivalent to Equation 3).

III. Methods

Three policies based on advantage are investigated in this paper. Advantage is defined as the marginal loss in value of taking an arbitrary action compared to the optimal action:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s) \tag{5}$$

Note that optimal action(s) has an advantage of 0, while all other actions have negative advantages. In the same way that a policy can be found from the Q function, it can be found from the A function since $A(a, s)$ and $Q(a, s)$ differ by a constant for all a at a given s:

$$\pi(s) = \arg\max_a Q^\pi(s, a) = \arg\max_a A^\pi(s, a) \tag{6}$$

A. Prior Benchmarks

In a concurrent paper [15], a semi-analytical method of estimating advantages in this problem is investigated. While the methods developed are outside the scope of this paper, two relevant results are used here:

1) Near-Optimal Policy: A near-optimal policy was constructed that serves as a lower bound on the maximum reward for the problem. In short, this method works by constructing an approximate directed acyclic graph of possible slews, then searches the graph for a maximum-value path. Note that this policy is supplied a larger set of upcoming targets $U_{M'}$ in the state than the other policies: $M' > M$. Due to the combinatorial nature of the problem, this policy is computationally expensive but produces relatively high-quality target sequences. It takes roughly 20 seconds to recompute at each decision interval.

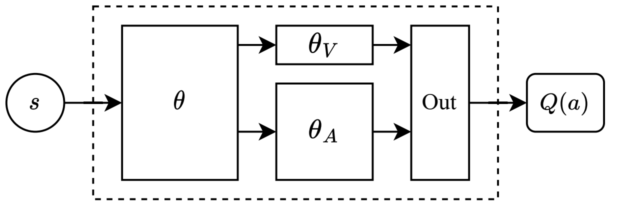

Fig. 2: The dueling DQN architecture with network parameters θ, the θ_V, and θ_A; the value V and advantage A branches are combined using Equation 7.

2) Advantage Heuristic: In finding the near-optimal policy, good estimates of advantage are produced. Since the near-optimal policy is expensive to evaluate, the advantage estimates are regressed over, resulting in an advantage heuristic $A^h(s,a)$ that can be computed cheaply with a single network evaluation. Even though A^h is an imperfect regression over already imperfect estimates of advantage, it can still be used as a reasonably effective policy via Equation 6. For comparison with days of computation time needed to train the RL algorithms presented later, this heuristic network takes hours to generate synthetic data and train. The efficiency of the heuristic generation primarily comes by avoiding use of the computationally expensive simulation environment.

B. Dueling Deep Q-Learning

Dueling deep Q-learning is an alternative network architecture for DQN that separates value and advantage estimation into separate branches of the Q network [16]. The two branches of the network, as shown in Figure 2, are combined into a vector of Q estimates using

$$Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s,a') \right) \quad (7)$$

While subtracting the maximum $A(s,a')$ rather than the average would retain the proper semantics for advantage, Wang et al. note that the averaging operator yields better performance. To update the parameters of the Q network, the standard DQN loss function [17] is used:

$$L(s,a,r,s') = \left(r + \gamma \max_{a'} Q_{\theta'}(s',a') - Q_\theta(s,a) \right)^2 \quad (8)$$

Dueling DQN was selected as the domain knowledge-free learning RL method for this research primarily because it separates advantage from value estimations, resulting in a network architecture most comparable to the domain knowledge-informed method described in the next section. The dueling DQN architecture also tends to learn faster than standard DQN, which is advantageous due to the computationally expensive nature of the simulation environment.

C. Advantage Error Q-Learning

The objective of the domain knowledge-informed policy is to use RL to improve upon the policy produced by the advantage heuristic A^h. The advantage heuristic error is defined as

$$\delta A(s,a) = A^*(s,a) - A^h(s,a) \quad (9)$$

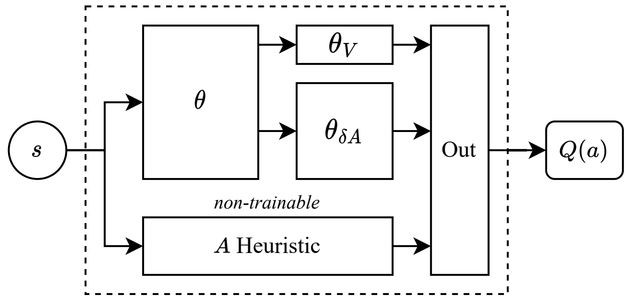

Fig. 3: The advantage error-learning architecture; the value V and advantage error δA branches are combined with the advantage heuristic A^h using Equation 10.

the error between the optimal and heuristic advantage functions. By substituting $A = A^h + \delta A$ into Equation 7, the network architecture in Figure 3, with a Q function defined as

$$Q(s,a) = V(s) + \Bigg(A^h(s,a) + \delta A(s,a) - \frac{1}{|\mathcal{A}|} \sum_{a}^{\prime} A^h(s,a') + \delta A(s,a') \Bigg) \quad (10)$$

As with dueling DQN, this formulation has the advantage of only modifying the Q network architecture and not the DQN algorithm. This ensures parity between the implementation of domain knowledge-informed and -free methods used.

D. Problem Parameters

Settings for the simulation environment and training algorithm are explained:

1) Environment Setup: To reduce the scope of the project, the environment was restricted to a fixed $N_T = 5000$ global targets, which translates to roughly 500 targets in view of the three-orbit ground track. The state space was further reduced by restricting the satellite to a fixed 3-orbit trajectory; in practice, one would train over all longitudes for any inclination angles of interest.

The discount factor γ is reformulated to be in terms of the time between decision intervals rather than number of decision intervals

$$\gamma(s,s') = \gamma^{t_{s'} - t_s} \quad (11)$$

so that taking more actions in a set timespan does not overpenalize actions toward the end of the timespan. This modification does not lose any MDP properties, as it can be interpreted an MDP with an arbitrarily fine fixed timestep with zero reward at all intermediate timesteps. The loss function then becomes equivalent to a multistep lookahead to the next non-zero reward by modifying Equation 8 with Equation 11:

$$L(s,a,r,s') = \left(r + \gamma(s,s') \max_{a'} Q_{\theta'}(s',a') - Q_\theta(s,a) \right)^2 \quad (12)$$

A very small $\gamma = 0.99^{1/250\text{s}}$ is used because discounting does not have any physical meaning in the problem, but a non-unity γ showed better stability in training even with a finite-horizon problem.

2) Network Hyperparameters: The networks are constructed with an output vector length of $N = 15$ making the reasonable assumption that at the chosen target density, the optimal next target is almost always within $U_1 5$, the 15 upcoming targets. The input state considers the next $M = 20$ targets.

The primary network θ uses the same architecture for dueling DQN and advantage error learning, selected to be the same size as the heuristic network so that a reasonable level of expressivity can be assumed. These networks are all fully-connected, with 3 layers of 50 nodes using the GELU activation function. The branch networks in dueling DQN and advantage error learning are also the same size in both architectures: θ_V is 2 layers $\times$ 6 nodes, and θ_A is 1 layer $\times$ 20 nodes.

3) Training Hyperparameters: The DQN algorithm used by both RL methods is implemented in Python, using TensorFlow [18] for neural networks. Because of the computational expense of the simulator, simulations are run in batches such that each epoch simulated multiple episodes simultaneously. Memory limitations on the machine limit the maximum episodes per epoch to 20. The Q network is trained on approximately 10% of the experience tuples from an experience buffer composed of the prior 30 epochs. The network is optimized with the Adam optimizer. The fixed θ' network in DQN is updated every 15 epochs. An ϵ-greedy policy is used during training, with

$$\epsilon(\text{epoch}) = \max(0.1, 1 - \frac{\text{epoch}}{15}) \tag{13}$$

Each network is trained for 1,000 epochs, or 20,000 episodes. Since each episode typically yields 100 to 150 experience tuples, this training translates to 2-3 million experience tuples total. With multiprocessing, this takes 3 to 4 days of training on an AMD Ryzen Threadripper.

IV. Results

A. Learning Curves

For each RL algorithm, the reward, average per-target reward $\bar{\rho}$, number of targets successfully imaged, and success percent of imaging attempts are examined over the course of training. These values are compared to the corresponding values for a random policy as a baseline and for the heuristic policy as a target to exceed.

1) Dueling DQN: Dueling DQN improves from a random policy, shown in Figure 4. At first, it achieves this by learning to select higher value targets, peaking at $\bar{\rho} \approx 0.6$ around 6000 episodes into training. However, learning then begins to favor increasing the number of images successfully taken at the expense of the value of the targets. While the reward curve remains relatively flat throughout the second half of training, the number of imaged targets continues to rise, implying that the policy potentially did not converge.

2) Advantage Error Learning: Advantage error learning shows a disappointing lack of development over the course of training. Once the ϵ ramp-down is complete, the policy's metrics tend to stay near the values of the heuristic policy. One interesting development occurs about halfway through training: the imaging success rate drops while the number of successful images stays relatively constant. This implies that the policy begins to more aggressively attempt targets, but is not able to translate that into an increased number of successes. This indicates that additional learning could lead to policy improvement.

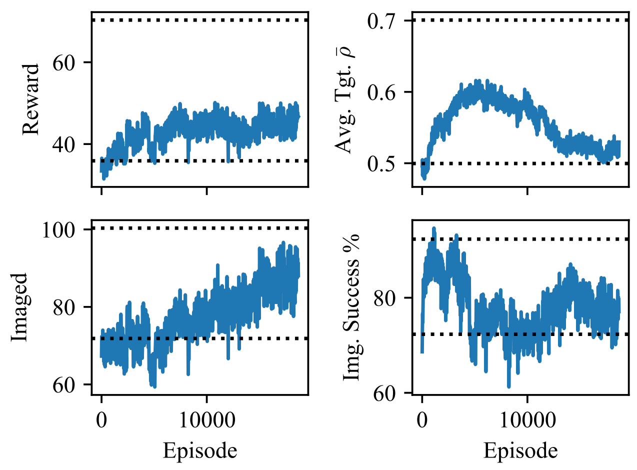

Fig. 4: Dueling DQN training curves. Curves report ϵ-greedy values. Random policy (lower) and heuristic (upper) values dotted for reference.

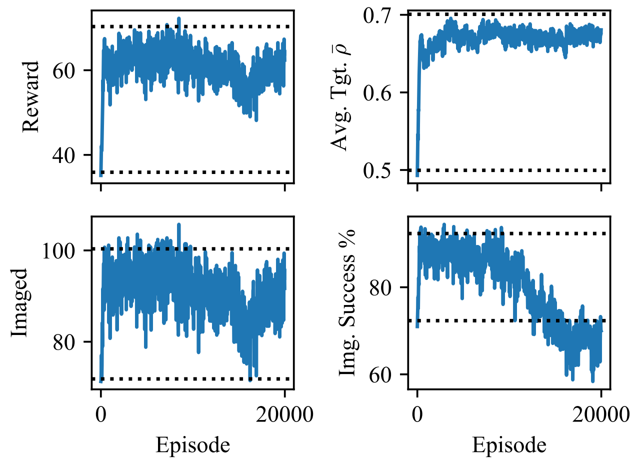

Fig. 5: Advantage error training curves. Refer to Figure 4 for additional notes.

B. Policy Performance Comparison

The two RL policies are compared to a random policy, the heuristic policy, and the near-optimal analytic policy in Figure 6 and Table I. In terms of reward, it is clear that dueling DQN was able to improve beyond a random policy, but neither RL policy exceeds the performance of the heuristic policy, even though the performance gap between the heuristic and the near-optimal policy demonstrates that there is considerable room for improvement on the heuristic.

	Random	**Dueling**	**Adv. Error**	**Heuristic**	**Near Opt.**
Reward	35.9 ± 0.6	48.2 ± 1.0	65.6 ± 1.1	70.3 ± 1.2	106.1 ± 2.7
Avg. Target $\bar{\rho}$	0.501 ± 0.004	0.527 ± 0.003	0.691 ± 0.002	0.701 ± 0.003	0.757 ± 0.003
Imaged	71.9 ± 1.1	91.3 ± 1.8	95.0 ± 1.5	100.3 ± 1.7	140.3 ± 3.6
Attempted	99.4 ± 0.9	117.0 ± 0.7	153.4 ± 1.2	108.7 ± 0.8	149.3 ± 2.0
Img. Success %	72.2 ± 0.9	77.5 ± 1.3	61.8 ± 0.8	91.7 ± 1.0	92.8 ± 1.4

TABLE I: Comparison of performance metrics across different policies (mean $\pm$ standard error).

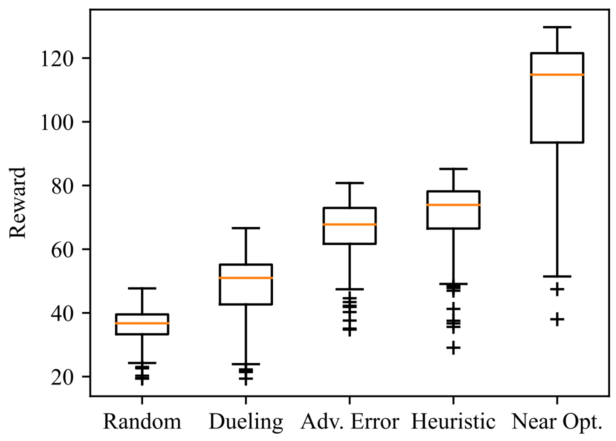

Fig. 6: Reward distribution across 100 episodes for each policy.

A comparison of other metrics reinforces observations made about the training curves. Dueling DQN's greatest shortcoming is poor target selection with an average target value only marginally above that of the random policy. In the case of the advantage error network, its imaging success rate is very low even compared to the random policy, attempting as many images as the near-optimal policy but succeeding less than two-thirds of the time. However, this poor behavior only marginally decreases the other metrics relative to the heuristic. For both RL algorithms, it is difficult to conclusively understand their performance given the potential for additional learning and fact that analyzing the strategies from a human perspective is not necessarily meaningful.

V. Conclusion

While the dueling DQN results demonstrated that learning target sequencing in the AEOSSP is feasible, the poor performance of both RL methods compared to a good but imperfect heuristic limits the utility of the solution. This is especially true when considering the generation time for the heuristic is — at worst — less than 10% of the learning time for the RL policies. However, these results encourage considerable future work on this topic:

- Various indications that training was not complete were present in the training curve results. While a longer training time was outside the scope of feasibility for this project, better computing resources and time could show improvement beyond the heuristic.
- It is possible that the heuristic network architecture reached maximum expressivity with respect to the state vector. Further investigation of network hyperparameters and state vector sizes (e.g. increasing M in U_M) may yield better policies with less training.
- Reward function tuning may also improve the training rate. In this problem, it can be reasonably assumed that an optimal policy will never fail imaging. While not in line with the natural semantics for reward, a large negative reward could be returned in response to failed images; this may discourage the negative success rate trajectory displayed in the training curve in Figure 5.
- RL algorithms such as PPO, which has performed favorably in similar problem formulations [10], could be applied to the problem. Currently, an improved simulation environment is under development that would allow better compatibility with Stable Baselines implementations of these algorithms, enabling such experiments.

If improvements in training are achieved through any of the proposed modifications, the next step would be to explore generalizing the policy to all target densities and orbits.

A positive corollary to the negative result in this paper is that the handcrafted policies proposed in an upcoming paper [15] yield a performance that is nontrivial to achieve by other methods.

VI. Contributions and Release

Mark Stephenson is the sole contributor to the work presented in this paper. Adam Herrmann developed the original simulation environment this environment is based on. Code can be found in the repository at bitbucket.org/avslab/basilisk-gym-interface/, with code specific to this project present on the branch "feature/VariableIntervalTraining".

The author grants permission for this report to be posted publicly.

References

[1] 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, no. 3, pp. 3881–3892, Sep. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9115252/

[2] D.-H. Cho, J.-H. Kim, H.-L. Choi, and J. Ahn, "Optimization-Based Scheduling Method for Agile Earth-Observing Satellite Constellation," *Journal of Aerospace Information Systems*, vol. 15, no. 11, pp. 611–626, Nov. 2018. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.I010620

[3] J. Kim, J. Ahn, H.-L. Choi, and D.-H. Cho, "Task Scheduling of Agile Satellites with Transition Time and Stereoscopic Imaging Constraints," *Journal of Aerospace Information Systems*, vol. 17, no. 6, pp. 285–293, Jun. 2020. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.I010775

[4] 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, p. 107292, Jun. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0360835221001960

[5] V. Gabrel, A. Moulet, C. Murat, and V. T. Paschos, “A new single model and derived algorithms for the satellite shot planning problem using graph theory concepts,” *Annals of Operations Research*, vol. 69, pp. 115–134, 1997.

[6] C. Verbeeck, K. Sörensen, E.-H. Aghezzaf, and P. Vansteenwegen, “A fast solution method for the time-dependent orienteering problem,” *European Journal of Operational Research*, vol. 236, no. 2, pp. 419–432, Jul. 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0377221713009557

[7] G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, and P. Vansteenwegen, “Agile earth observation satellite scheduling: An orienteering problem with time-dependent profits and travel times,” *Computers & Operations Research*, vol. 111, pp. 84–98, Nov. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0305054819301510

[8] L. Wei, Y. Chen, M. Chen, and Y. Chen, “Deep reinforcement learning and parameter transfer based approach for the multi-objective agile earth observation satellite scheduling problem,” *Applied Soft Computing*, vol. 110, p. 107607, Oct. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1568494621005287

[9] M. Piccinin, P. Lunghi, and M. Lavagna, “Deep Reinforcement Learning-based policy for autonomous imaging planning of small celestial bodies mapping,” *Aerospace Science and Technology*, vol. 120, p. 107224, Jan. 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1270963821007343

[10] A. Herrmann and H. Schaub, “A Comparison Of Deep Reinforcement Learning Algorithms For Earth-Observing Satellite Scheduling,” in *AAS/AIAA Spaceflight Mechanics Meeting*, Austin, TX, Jan. 2023.

[11] A. Herrmann, M. Stephenson, and H. Schaub, “Reinforcement Learning For Multi-Satellite Agile Earth Observing Scheduling Under Various Communication Assumptions,” Breckenridge, CO, Feb. 2023.

[12] L. Bauersfeld, E. Kaufmann, P. Foehn, S. Sun, and D. Scaramuzza, “NeuroBEM: Hybrid Aerodynamic Quadrotor Model,” in *Robotics: Science and Systems XVII*, Jul. 2021, arXiv:2106.08015 [cs]. [Online]. Available: http://arxiv.org/abs/2106.08015

[13] 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, Sep. 2020. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.I010762

[14] D. Eddy and M. Kochenderfer, “Markov Decision Processes For Multi-Objective Satellite Task Planning,” in *2020 IEEE Aerospace Conference*. Big Sky, MT, USA: IEEE, Mar. 2020, pp. 1–12. [Online]. Available: https://ieeexplore.ieee.org/document/9172258/

[15] M. Stephenson and H. Schaub, “Advantage Learning For Autonomous Target Selection In The Agile Earth-Observing Satellite Problem,” in *AAS/AIAA Astrodynamics Specialist Conference*, Big Sky, Montana, Aug. 2023.

[16] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, “Dueling Network Architectures for Deep Reinforcement Learning,” Apr. 2016, arXiv:1511.06581 [cs]. [Online]. Available: http://arxiv.org/abs/1511.06581

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with Deep Reinforcement Learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” Mar. 2016, arXiv:1603.04467 [cs]. [Online]. Available: http://arxiv.org/abs/1603.04467