Learning Robust Policies for Generalized Debris Capture with an Automated Tether-Net System

Chen Zeng* , Grant Hecht* , Prajit KrisshnaKumar* , Raj K. Shah† , Souma Chowdhury‡ , Eleonora M. Botta§
University at Buffalo, Buffalo, NY, 14260

***Abstract*—Tether-net launched from a chaser spacecraft provides a promising method to capture and dispose off large space debris in orbit. This tether-net system is subject to several sources of uncertainty in sensing and actuation that affect the performance of its net launch and closing control. Earlier reliability based optimization approaches to design control actions however remain challenging and computationally prohibitive to generalize over varying launch scenarios and target (debris) state relative to chaser. To search for a general and reliable control policy, this paper presents a reinforcement learning framework that integrates a proximal policy optimization (PPO2) approach with net dynamics simulations. The latter allows evaluating the episodes of net-based target capture, and estimate the capture quality index that serves as the reward feedback to PPO2. Here, the learnt policy is designed to model the timing of the net closing action based on the state of the moving net and the target, under any given launch scenario. A stochastic state transition model is considered in order to incorporate synthetic uncertainties in state estimation and launch actuation. Along with notable reward improvement during training, the trained policy demonstrates capture performance (over a wide range of launch/target scenarios) that is close to that obtained with reliability based optimization run over an individual scenario.**

Keywords: Active Debris Removal, Reinforcement Learning, Tether-Net, Uncertainty

I. Introduction

Active Debris Removal (ADR) using tether-net systems has been proposed and studied as a promising solution to the space debris problem [1], [2], [3], [4], [5], [6], [7]. Among others, Botta et al. [8], [9], [10] conducted extensive research on the dynamics of the deployment and capture phases of net-based debris removal missions.

The majority of existing works are based on the assumptions of capturing a specific target with ideal launch conditions, with a handful of pioneering works (Salvi [11], Botta et al. [9], [8], and Endo et al. [12]) looking into the robustness of deployment or capture under various net launch conditions in the absence of a closing mechanism. In this type of mission, it is crucial to guarantee a successful capture of the target in the presence of uncertainties, and to ensure that the target remains wrapped by the net when the chaser tugs it to its disposal orbit. These uncertainties can be attributed to measurement errors, inaccuracies in net launch control, and to the estimation of the debris and chaser vehicle inertial and attitude states. However, to date, very little work has been performed to study the effects of uncertainties on the system robustness, especially in the presence of a closing mechanism, with the exception of work by Chen et al. [13].

The tether-net capture system presents a multidisciplinary robust design-control problem with pertinent constraints under uncertainties. In our preceding research[14], a reliability-based design optimization process was proposed to optimize the launching and/or closure of the net under the influences of uncertainties for a known fixed debris target. This process models uncertainties and performs Bayesian optimization to determine launch strategies that maximize the capture success rate. However, such a method only searches for a single strategy that applies to a predefined debris status, meanwhile requiring considerable computing power. Therefore, while it provided valuable insight on robust debris capture under uncertainty, this approach is ultimately not suitable as a flexible and flight-worthy solution.

Artificial Neural Networks (ANN) are playing an emerging role as decision-support models in various intelligent autonomous systems [15]. As a universal function approximator, an ANN is capable of mapping states to actions in autonomous systems, a.k.a. a policy model. Various ANN fitting (learning) methods have seen demonstrations on robotics and control applications. Popular learning methods include Reinforcement Learning[16], [17], Supervised Learning[18], Imitation Learning[19], Neuroevolution[20], [21], etc., among which the advanced reinforcement learning[22] and neuroevolution[23] methods are directly applicable to launching and wrapping control of tether-net systems. These aforementioned machine learning methods bring capabilities of adapting to system uncertainties, and selecting optimal actions (policies) according to various debris characteristics. Computation-heavy reliability analyses required in the optimizations can also be reduced, retaining the computing load similar to that of reliability-based optimizations.

This paper proposes a machine-learning-based policy optimization of a one-shot robotics system with environment adaptability and robustness under uncertainties. The case study features the launching and wrapping process for a tether-net

* Ph.D. Student, Department of Mechanical and Aerospace Engineering
† M.S. Student, Mechanical and Aerospace Engineering
‡ Associate Professor, Mechanical and Aerospace Engineering, Corr. author email: soumacho@buffalo.edu
§ Assistant Professor, Mechanical and Aerospace Engineering

**This work was supported by the NSF award CMMI 2128578. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

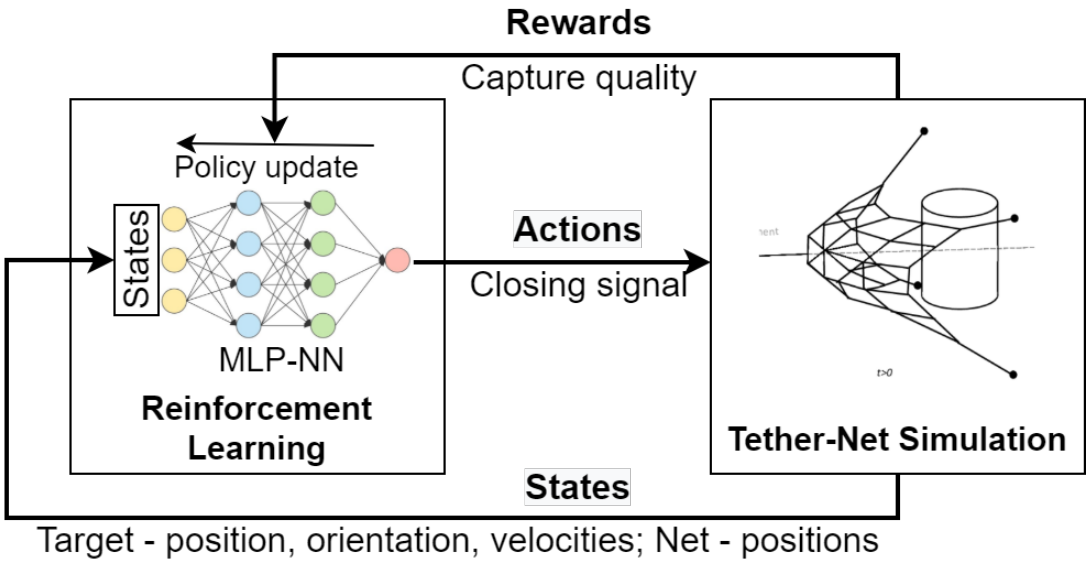

Fig. 1: Proposed Policy Learning Process for the Tether-Net System

space debris capture system. The machine learning framework adapts to various environmental parameters (including net geometry and states of the chaser and the target), considers multiple sources of uncertainties (including inaccuracies of state estimation, errors in launching and wrapping parameters, and sensing errors), and determines the optimal wrapping parameters that maximize the probability of a successful capture. Environment parameters are relative to the capture system (e.g., the mass of the corner masses and the geometry of the net) and the state of the debris (e.g., the distance to the chaser, its orientation, and its motion). The wrapping policy comprises only triggering of the closing mechanism. The capture success is evaluated by the Capture Quality Index, or CQI (interpreted later in the manuscript). A case study scenario is presented with standalone wrapping policy learning with programmed launching employing a state-of-the-art learning technique, Proximal Policy Optimization 2 (PPO2)[22].

The remainder of this paper is organized as follows: In Section II, the architecture of the simulator is presented briefly, together with the models implemented for the different components of the system. In Section III, the adopted machine-learning-based policy optimization is discussed. Section IV presents validation of the optimization results, and Section V concludes the paper with a discussion of results and limitations of the work, and associated planned future work.

II. Modeling: Tether-Net Launching and Wrapping Mechanics

Inherited from the preceding work[10], [24], [14], the modeled system comprises of a chaser carrying a square-shaped net with 4 corner masses and a closing mechanism around the perimeter. The tether-net system is simulated in Vortex Studio, a powerful multibody dynamics simulation platform designed for real-time simulation of complex mechanics. The net is modeled with the standard lumped-parameter approach. The mass properties of the net are lumped into small spherical rigid bodies placed at the physical knots of the net, herein referred to as nodes. The axial stiffness and damping properties of the net's threads are represented with massless springs and dampers in parallel between the nodes.

TABLE I: Design and properties of the tether-net system

Variable	Description	Value
L	Side Length of Net	22.0 m
$L_{\mathtt{mesh}}$	Side Length of Mesh	1.375 m
r_{n}	Thread Radius	6 mm
m_{n}	Total Mass of Net	100 kg
m_c	Mass of Corner Mass	10.0 kg
$\mathbf{V}_{\mathtt{eb}}$	Launching Velocity	$[3.30, 3.54, 7.16]$ m/s

The chaser spacecraft is modeled as a cubic rigid body in the simulation. The main tether, linking the net to the chaser, is modeled with a series of slender rigid bodies, modeled as relaxed prismatic joints to simulate the axial and bending stiffness and damping properties.

The closing mechanism applies a drawstring interlaced with the perimeter of the net, as shown in Fig. 2. The drawstring passes through 8 nodes on the perimeter as well as the 4 corner masses, and is winched independently from the main tether[10]. When the closing mechanism is activated, constant forces are made to act between each pair of adjacent nodes along the drawstring, pulling the nodes together until contact. Upon contact, the node pairs are locked to keep the mouth of the net closed for the rest of the maneuver.

The detailed design of the tether-net system is fixed as the derived result from preceding work[14]. Table I lists the design parameters.

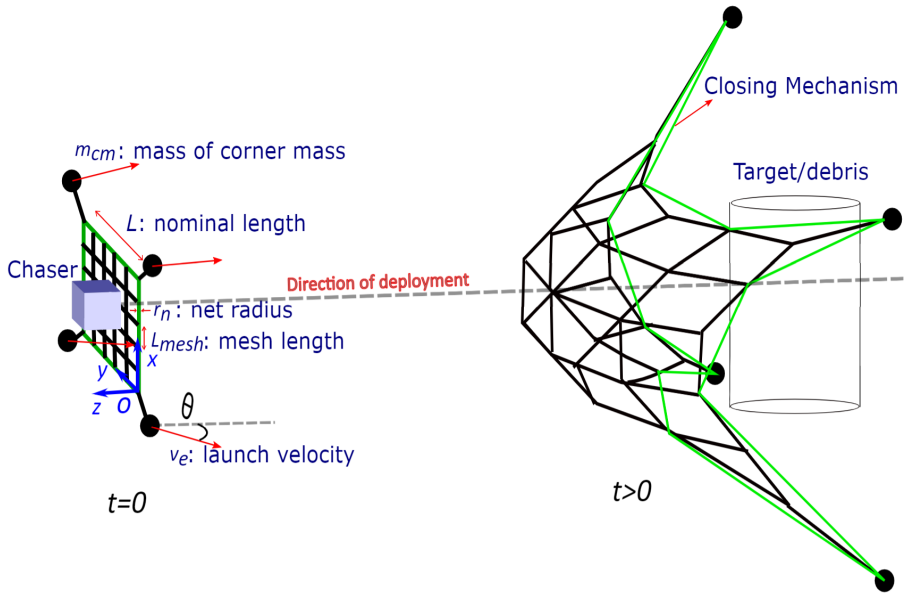

Fig. 2: Sketch of the Modeled Tether-net System

III. Learning the Optimal Launching and Closing Policies

A. Defining the learning task

Reinforcement learning models the actions as Markov Decision Processes [25] (MDP) that comprise a comprehensive state space with a relatively compact action space. There are 2 sets of actions (launching and closing) taking effect on different stages of the capture operation, we are focusing on closing within the scope of this paper.

The policy model of the closing signal is time variant, while the action itself is a logical variable (boolean). The observations (state space) are assumed to be estimated by employing readily available sensors, including: 1) internal measurement units (IMUs), cameras, or Lidar mounted on the chaser vehicle, 2) IMUs and cameras attached to the corner masses, as well as 3) monitoring from the Earth. Therefore, the state space mainly consists of the position, orientation, and velocities of the target, as well as the positions of the corner masses of the net. In addition, the velocity of launching the corner masses, the simulation time, and a flag indicating the actuation of closing are also part of the observation. Some parameters, like the position of the net's center of mass, are obtained from the simulation for calculations of the reward and the constraints, but excluded from the state space since they are unattainable with the conceived sensors. Table II lists the state space and the action space parameters.

The objective of reinforcement learning is to train the policy model to find the optimal timing of sending the closing signal, based on observations regarding the states of the net and the target. The policy model adapts a reward function defined as:

$$\max_{\mathbf{Q}}: \quad R_A = \left(\sum_{t_0}^{t_{\mathtt{close}}} r(t) + r_{\mathtt{end}} \right) / n_{\mathtt{steps}} \tag{1a}$$

$$\text{where:} \quad r(t) = \begin{cases} w_1 \cdot \left(C_1 - \left|\frac{V_{\mathrm{n}} - V_{\mathrm{t}}}{V_{\mathrm{t}}}\right|\right) + w_2 \cdot \left(C_2 - \left|\frac{S_{\mathrm{n}} - S_{\mathrm{t}}}{S_{\mathrm{t}}}\right|\right) + w_3 \cdot \left(C_3 - \left|\frac{q_{\mathrm{n}}}{q_{\mathrm{t}}}\right|\right) + w_4 \cdot (N_{\mathrm{L}} - C_4) + \dots \\ \qquad 0.4 \cdot (\min(t, t_1) - 4.6) - (0.12 \cdot (\max(t, t_2) - t_2))^2 + C_5 \quad \text{(no premature closing)} \\ -(t_1 - t_{\mathtt{close}})^2 \qquad \text{(with premature closing)} \end{cases} \tag{1b}$$

$$r_{\mathtt{end}} = \begin{cases} w_1' \cdot \left(C_1' - \left|\frac{V_{\mathrm{n}}' - V_{\mathrm{t}}}{V_{\mathrm{t}}}\right|\right) + w_2' \cdot \left(C_2' - \left|\frac{S_{\mathrm{n}}' - S_{\mathrm{t}}}{S_{\mathrm{t}}}\right|\right) + w_3' \cdot \left(C_3' - \left|\frac{q_{\mathrm{n}}'}{q_{\mathrm{t}}}\right|\right) + \dots \\ \qquad w_4' \cdot (N_{\mathrm{L}}' - C_4') + C_5' \quad \text{(closing started before 60 s)} \\ C_6 \qquad \text{(not closing by 60 s)} \end{cases} \tag{1c}$$

In which $\mathbf{Q}$ represents the policy model; R_A represents the

TABLE II: Parameters of State and Action Spaces

Type	Variable	Data Type	Boundaries
State	Time	Scalar	0 to 120 s
	Target Coordinates ($p_{\texttt{target}}$)	Cartesian	$[-10, -10, 0]$ to $[10, 10, 50]$ m
	Target Orientation	Euler angles	$[0, 0, 0]$ to $[2\pi, 2\pi, 2\pi]$ rad
	Target Angular Velocity	Euler angles	$[-1, -1, -1]$ to $[1, 1, 1]$ rad/s
	Pos. of Corner Masses (4)	Cartesian	$[-22, -22, 0]$ to $[22, 22, 72]$ m
	Closure Flag (f_c)	Boolean	-
	Launching Velocity	Cartesian	$[1, 1, 1]$ to $[5, 5, 10]$ m/s
Action	Closing Signal	Boolean	0 and 1

episodic (mean) reward; t represents the simulation time; t_0 stands for the initial and final time steps; $t_{\texttt{close}}$ represents the time when the closing signal is issued; t_1 and t_2 are manually configured time triggers; $n_{\texttt{steps}}$ stands for the number of (learning) steps in the episode; $r(t)$ represents the reward at time step t; $r_{\texttt{end}}$ represents the bonus end-of-episode reward; V_{n}, S_{n}, and q_{n} stand for the enclosed volume, surface area, and the center-of-mass position of the net at time t; V'_{n}, S'_{n}, and q'_{n} stand for those at time ($t_{\texttt{close}} + 20$ seconds); V_{t}, S_{t}, and q_{t} stand for the volume, surface area, and the center-of-mass position of the target; N_{L} is the number of locked node-pairs around the edge (12 pairs in total); C_1 through C_6, as well as w_1 through w_4 are tuning weights, in addition to C'_1 through C'_5 and w'_1 through w'_4; f_C refers to the closure flag.

A premature closing is defined as:

$$\begin{cases} \textbf{True} & \text{if } (f_C = 1) \cap (t_{\texttt{close}} < 15 \text{ s}) \cap \dots \\ & [(q_{\text{n}}(t_{\texttt{close}}) > 12 \text{ m}) \cup (|\overline{p}_{\texttt{cm}} - p_{\texttt{target}}| > 10 \text{ m})] \\ \textbf{False} & \text{otherwise} \end{cases} \tag{2}$$

where $\overline{p}_{\texttt{cm}}$ represents the mean position of the corner masses, and $p_{\texttt{target}}$ represents the position of the target.

A crucial part of the reward is largely dependent on the distances between the net and the target in terms of position, surface area, and volume. Such formulation is inspired by the Capture Quality Index (CQI)[26], which estimates the successfulness of a capture with the aforementioned measurements. The number of locked node-pairs (N_{L}) indicates the possibility of the target slipping out of the net, and is also considered in the formulation for added robustness.

The conditional formulations in the reward function ensures some substantial penalties for premature closing (too far away from the target) or delayed closing (too late), and a substantial bonus (C'_5) for appropriately-timed closing. Timing is vital for a successful capture, but the time-variant nature of the closing signal makes it difficult for the policy model to learn to avoid premature action. A reward function solely based on the CQI is insufficient to distinguish between an obvious failure and a possibly successful scenario, hence the added penalty that overrules the reward is necessary to avoid possible exploitation of the CQI formulation. Since the reward function returns positive rewards through the majority of an episode, the policy model could exploit the formulation by not closing at all. The bonus reward (C'_5) for well-timed closing and the penalty (C_6) for not closing at all would work together to prevent the policy model from refusing to close throughout the episode.

We train the policy model in stages controlled by step count, while adjusting the tuning weights as the stages progress. The CQI part of the end-of-episode reward $r_{\texttt{end}}$ is gradually magnified as the policy model learns to avoid premature closing. Penalty for delayed closing (C_6) is inactive until the policy model learns to refuse to close. Table III lists the values of the adjustable parameters applied through various stages of training.

Two scenarios were conceived: 1) learning a standalone wrapping policy while launching is programmed; 2) learning launching and wrapping policies simultaneously. Scenario 2 calls for training two policy models in one simulation, which brings in questions like reward crediting [27] and the orders of execution. Limited by time and computing constraints, the case study in this paper is confined to scenario 1). The programmed launching policy for scenario 1 was presented in a previous work [14] (values listed in Table I).

B. Design of experiments and uncertainty modeling

The Design of Experiments samples a range of the target's distance, initial orientation, and initial rotating angular velocity. Every episode of training is sampled with random choices of the forementioned initial states of the target. Table IV lists the variations of DoE.

The Simulation cannot directly simulate the onboard sensors and actuators in high fidelity, therefore uncertainties are modeled in sensing, state estimation, and actuation of the system by applying stochastic noises to the parameters. Sources of uncertainties include: 1) estimated state of the debris target and the corner masses of the net, 2) velocity of launching the net, 3) timing of triggering the closing mechanism, and uniquely 4) soft dynamics of the net. The uncertainty of the net dynamics is aleatoric and unavoidable, while the majority of the uncertainties in estimation and control are epistemic [28].

Gaussian noises modeled after Table V are applied upon the observed parameters from the simulation before delivery to the learning algorithm. Noises of the constant parameters

TABLE III: Coefficients for the Reward Formulation

Unchanged	w_1	w_2	w_3	w_4	C_1	C_2	C_3	C_4	C_5	t_1	t_2
	0.025	0.025	0.2	0.125	3	3	5	2	2	15	20
	w_1'	w_2'	w_3'	w_4'	C_1'	C_2'	C_3'	C_4'	C_5'	C_6	
0 to 66,000 steps	0.05	0.05	0.4	0.125	3	4	6	0	50	0	
66,001 to 300,000 steps	0.1	0.1	0.8	0.25	3	3	6	0	50	0	
300,001 to 800,000 steps	1	1	8	2.5	3	3	6	0	50	-50	
800,001 to 1,500,000 steps	2	2	16	5	3	3	3	2	100	-50	

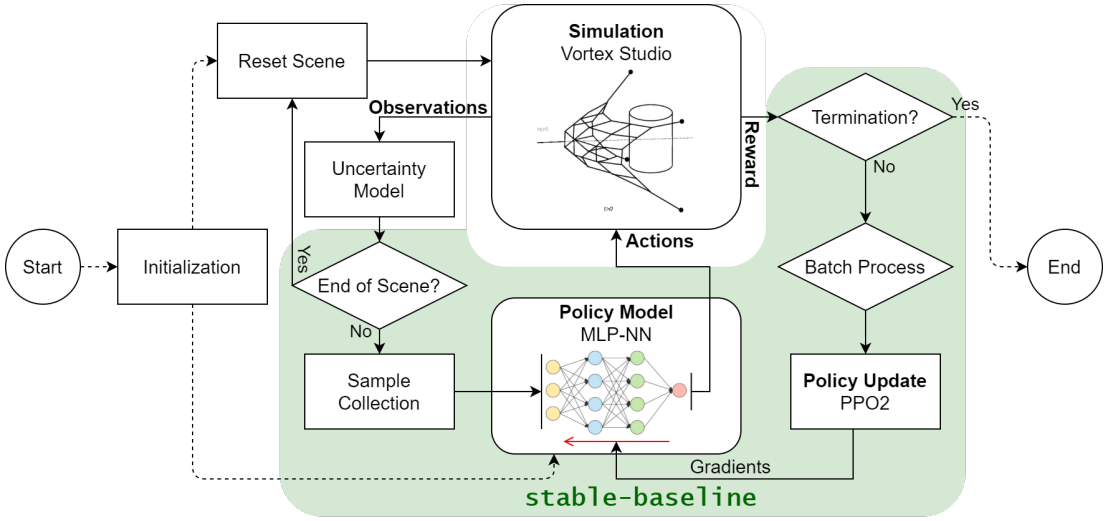

Fig. 3: The Workflow of Reinforcement Learning on the Tether-Net Capture Mission

TABLE IV: Initial States of the Target

Name	Minimum	Maximum
Dist. to chaser	25 m	35 m
Orientation	$[0, 0, 0]$ rad	$[\frac{\pi}{2}, 0, 0]$ rad
Angular Vel.	$[0, -\frac{\pi}{18}, -\frac{\pi}{18}]$ rad/s	$[0, \frac{\pi}{18}, \frac{\pi}{18}]$ rad/s

(target orientation and launching velocity) are one-shot, while those of the continuously monitored parameters are sampled at every time step of learning.

C. Learning and validation techniques

Herein, the learning technique known as Proximal Policy Optimization 2 (PPO2) from stable baselines [29] is applied within a case study to provide an opportunity to evaluate and compare advanced reinforcement learning with prior results obtained by robust optimization under a fixed environment [14]. PPO2 is a state-of-the-art actor-critic reinforcement learning method which has demonstrated high efficiency, wide adaptability, and robust reliability [30].

PP02 employs a gradient update based on the experiences collected on the mini-batch of interactions with the environment. Upon update completion, the experiences collected are then discarded and the next update is based on new experiences. PPO2 has been proven to demonstrate less variance during the training process when compared to alternative learning techniques which ensures a smoother training process [31]. A multi-layer perceptron model consisting of 2 layers with 64 neurons is used within the deep Q network. This network is trained for 500 episodes and the trained policy is then validated for 100 episodes. Parameters used for the learning process are provided in Table VI.

For the final evaluation, a reliability sampling process was performed to examine the probability of success for the trained policy model. This involved a Monte Carlo sampling process to sample the impact of the uncertainties, in which $N = 100$ is number of independent simulations executed with the trained policy model. The evaluated probabilities of success are then compared to the success rate achieved by robust optimization under a fixed environment[14].

IV. Results

The learning trials were executed on a Windows workstation with 16 CPU cores with parallel computing with 30 workers for the episode evaluations. The learning process progresses in 20,000-to-500,000-step stages as we examine the learning rate and adjust the reward weights in between stages. By the conclusion of this paper, a total of 1.5 million steps of learning have been finished. The time cost of learning was 41.2 hours.

The history of episodic reward from one of the 30 workers is displayed in Figure 4. This specific worker finished 549

TABLE V: Noise Levels of Modeled Uncertainties

Sampled During:	Noise Source	Data Type	Margin of Error (2σ)
Step-wise	Target Orientation	3D Vector	$\pm[\pi/36, \pi/36, \pi/36]$ rad
	Target Angular Velocity	3D Vector	$\pm[\pi/36, \pi/36, \pi/36]$ rad/s
	Corner Masses Position	3D Vector	$\pm[0.1, 0.1, 0.25]$ m
Only Once	Target Position (CoM)	3D Vector	$\pm[0.1, 0.1, 0.25]$ m
	Launching Velocity	3D Vector	$\pm[0.05, 0.05, 0.1]$ m/s

TABLE VI: Reinforcement Learning Parameters

Algorithm	PPO2
Neural Network type	Multi-Layer Perceptron
Total Timesteps	1,500,000
Learning Rate	2.5e-4
Discount Factor	0.999
Number of Steps	128
Entropy Coefficient	0.01
Clip Range	0.2
Value Function Coefficient	0.5
Max. Gradient Norm	0.5

episodes in 46,437 steps. Since the episode tends to extend longer as the learning progresses, the later 89% of the steps only contributed to 80 episodes. The lowest episodic reward is -69.0, and the highest episodic reward is 9.7.

The episodic reward plot shows strong fluctuations, indicating the learning rate is unstable throughout the learning process and has yet to approach convergence. The 10-episode mean reward is also shown within the figure, which provides a more stable indication of the range and the trend of the reward values. The policy model received negative rewards throughout most of the episodes, but just managed to receive near-zero or positive rewards after 460 episodes. The trend of the rewards and the fluctuations are both signs of insufficient training. The episodic mean reward is unsuitable as a direct measurement of capture quality (for the successfully closed cases), considering the lengths of episodes fluctuate in a wide margin.

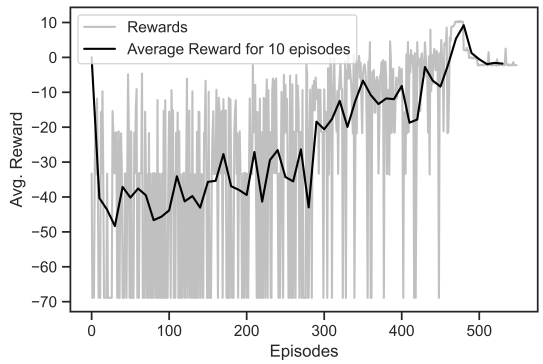

Fig. 4: History of Episodic Rewards

To evaluate the quality of the trained policy models, we tested all the neural networks obtained via different stages of learning with a Monte Carlo sample set sampled from the same initial states as shown in Table IV and applied the same level of uncertainties as shown in Table V. The best-performing policy model so far is from the end of 120,000 steps, which is compared to results from the previous work. The CQI values of the test result are calculated and compared with an optimized closing time under a static initial condition obtained from the predeceasing paper[14] (only closing was optimized for a single initial state). The mean CQI value of the policy model tests is 1.035, and the percentage of the samples with CQI values lower than 2 (seen as secure captures) is 94%. In contrast, the optimized fixed close timing managed to achieve 96% success rate and a mean CQI of 1.010 for the single initial state. The policy model of closing achieves a high success rate in a range of initial states, near that for the fixed close timing optimized for a single initial state, suggesting the policy model is approaching the maximal possible reliability. The policy model should keep improving if given more learning steps, and we expect the converged policy model to outperform the optimization results, especially in situations with large deviations.

V. Concluding Remarks

A machine-learning-based tether-net system wrapping policy optimization for ADR with environment adaptability and robustness under uncertainties was proposed and a trade study involving standalone wrapping policy learning with programmed launching employing a state-of-the-art learning technique, Proximal Policy Optimization 2 (PPO2), was performed.

Despite cutting off the learning process early, evaluation of the policy learning results shows that the proposed approach for wrapping policy learning proves promising, resulting in capture reliability comparable to earlier robust Bayesian optimization which involved a similar computational load, despite optimizing the wrapping strategy under a much wider range of state scenarios with larger uncertainty.

Future work will investigate standalone launching policy learning with programmed wrapping, and simultaneous learning of launching and wrapping policies. Additional machine learning tools, including reinforcement learning algorithms and advanced neruoevolution techniques[23] will also be applied for experiments.

REFERENCES

[1] Guang, Z. and Jing-rui, Z., "Space tether net system for debris capture and removal," *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, Vol. 1, IEEE, 2012, pp. 257–261.

[2] Wormnes, K., De Jong, J., Krag, H., and Visentin, G., "Throw-nets and tethers for robust space debris capture," *International Astronautical Congress*, 2013.

[3] Benvenuto, R., Salvi, S., and Lavagna, M., "Dynamics analysis and GNC design of flexible systems for space debris active removal," *Acta Astronautica*, Vol. 110, 2015, pp. 247–265.

[4] Botta, E., *Deployment and Capture Dynamics of Tether-nets for Active Space Debris Removal*, Ph.D. thesis, McGill University Libraries, 2018.

[5] Hausmann, G., Haarmann, R., Wieser, M., Brito, A., and Scheper, M., "OHB Concepts for Active Space Debris Removal and On-Orbit Servicing," *66th International Astronautical Congress, IAC*, 2015.

[6] Benvenuto, R. and Carta, R., *Implementation of a net device test bed for space debris active removal feasibility demonstration*, Ph.D. thesis, MS thesis, Politecnico di Milano, 2012.

[7] BOMBELLI, A., "Multidisciplinary design of a net-based device for space debris active removal," 2012.

[8] Botta, E. M., Sharf, I., and Misra, A., "Evaluation of net capture of space debris in multiple mission scenarios," *26th AAS/AIAA Space Flight Mechanics Meeting, Napa, CA*, AAS, 2016, pp. 16–254.

[9] Botta, E. M., Sharf, I., and Misra, A. K., "Energy and momentum analysis of the deployment dynamics of nets in space," *Acta Astronautica*, Vol. 140, 2017, pp. 554–564.

[10] Botta, E. M., Sharf, I., and Misra, A. K., "Simulation of tether-nets for capture of space debris and small asteroids," *Acta Astronautica*, Vol. 155, 2019, pp. 448–461.

[11] SALVI, S., "Flexible devices for active space debris removal: the net simulation tool," 2014.

[12] Endo, Y., Kojima, H., and Trivailo, P. M., "Study on acceptable offsets of ejected nets from debris center for successful capture of debris," *Advances in Space Research*, Vol. 66, No. 2, 2020, pp. 450–461.

[13] Chen, S., Woods, C. T., Boonrath, A., and Botta, E. M., "Analysis of the robustness and safety of net-based debris capture," *AIAA SCITECH 2022 Forum*, 2022, p. 1001.

[14] Kalpeshkumar Shah, R., Zeng, C., Botta, E., and Chowdhury, S., "Reliability-Based Launch and Closure Optimization for a Net-Based Space Debris Capture System," *2021 Multidisciplinary Analysis and Optimization Conference, AIAA AVIATION Forum (accepted)*, Jun 2021.

[15] Dounis, A. I. and Caraiscos, C., "Advanced control systems engineering for energy and comfort management in a building environment—A review," *Renewable and Sustainable Energy Reviews*, Vol. 13, No. 6-7, 2009, pp. 1246–1261.

[16] Baxter, J., Tridgell, A., and Weaver, L., "Knightcap: a chess program that learns by combining td (lambda) with game-tree search," *arXiv preprint cs/9901002*, 1999.

[17] Peters, J., Vijayakumar, S., and Schaal, S., "Reinforcement learning for humanoid robotics," *Proceedings of the third IEEE-RAS international conference on humanoid robots*, 2003, pp. 1–20.

[18] Caruana, R. and Niculescu-Mizil, A., "An empirical comparison of supervised learning algorithms," *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 161–168.

[19] Hussein, A., Gaber, M. M., Elyan, E., and Jayne, C., "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, Vol. 50, No. 2, 2017, pp. 1–35.

[20] Risi, S. and Togelius, J., "Neuroevolution in games: State of the art and open challenges," *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 9, No. 1, 2015, pp. 25–41.

[21] Hansen, N., Müller, S. D., and Koumoutsakos, P., "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary computation*, Vol. 11, No. 1, 2003, pp. 1–18.

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

[23] Behjat, A., Chidambaran, S., and Chowdhury, S., "Adaptive genomic evolution of neural network topologies (agent) for state-to-action mapping in autonomous agents," *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 9638–9644.

[24] Botta, E. M., *Deployment and capture dynamics of tether-nets for active space debris removal*, Ph.D. thesis, McGill University, Montreal, QC, 2017.

[25] Bellman, R., "A Markovian decision process," *Journal of mathematics and mechanics*, Vol. 6, No. 5, 1957, pp. 679–684.

[26] Barnes, C. M. and Botta, E. M., "A quality index for net-based capture of space debris," *Acta Astronautica*, Vol. 176, 2020, pp. 455–463.

[27] Nguyen, D. T., Kumar, A., and Lau, H. C., "Credit assignment for collective multiagent RL with global rewards," 2018.

[28] Hora, S. C., "Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management," *Reliability Engineering & System Safety*, Vol. 54, No. 2-3, 1996, pp. 217–223.

[29] Hill, A., Raffin, A., Ernestus, M., Gleave, A., Kanervisto, A., Traore, R., Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., and Wu, Y., "Stable Baselines," https://github.com/hill-a/stable-baselines, 2018.

[30] Ilyas, A., Engstrom, L., Santurkar, S., Tsipras, D., Janoos, F., Rudolph, L., and Madry, A., "Are Deep Policy Gradient Algorithms Truly Policy Gradient Algorithms?" *CoRR*, Vol. abs/1811.02553, 2018.

[31] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D., "Deep Reinforcement Learning that Matters," *CoRR*, Vol. abs/1709.06560, 2017.