Automated Trajectory Planning: A Cascaded Deep Reinforcement Learning Approach for Low-Thrust Spacecraft Orbit-Raising

Supplementary Material

1 Transfer Scenarios

The transfer scenarios presented in the paper not only differ in terms of the initial and final orbit but also vary due to the initial spacecraft parameters. These spacecraft parameters play a significant role in trajectory optimization problems. To ensure fair comparisons with other results in the literature, we standardized these parameters based on their values. As a result, we considered three Geostationary Transfer Orbit (GTO) initial orbit scenarios (GTO-1, GTO-2, GTO-3) and two Super-Geostationary Transfer Orbit (Super-GTO and Super-GTO-2) scenarios, as outlined in Table 1 in the main text. The details of these spacecraft parameters are presented in Table 1.

Scenarios	**Impulse $\mathbf{I}_{sp}$**	**Efficiency λ**	**Power P**	**Mass m**	**Thrust F**	**Thrust during shadows**
GTO-1 to GEO	1800s	55%	5kW	1200kg	0.31N	No
GTO-2 to GEO	3300s	65%	5kW	450kg	0.20N	No
GTO-3 to GEO	3000s	-	-	2000kg	0.35N	Yes
Super-GTO to GEO	3300s	65%	10kW	1200kg	0.40N	No
Super-GTO-2 to NRHO	1500s	-	-	1000kg	1N	Yes

Table 1: Spacecraft initial parameters across all transfer scenarios

2 CDRL Framework

In this section we will provide the additional details and calculations for our cascaded deep reinforcement learning framework.

2.1 State Elements

As introduced in the main paper in Section 4.3, our Cascaded Deep Reinforcement Lerning (CDRL) aprroach utilized the five 'he' elements as a state vector which are shown as follows:

$$\mathbf{s} = \begin{bmatrix} h & h_X & h_Y & e_X & e_Y \end{bmatrix}^T \tag{1}$$

where h denotes the magnitude of angular momentum, while h_X and h_Y denote the component of specific angular momentum along the Earth-centered inertial reference frame and the components of the eccentricity vector e_x and e_y in a non-inertial reference frame obtained after a 2-1 Euler rotation sequence. The initial values of these elements for all scenarios are stated in Tab 2.

2.2 Convergence Parameters

In Section 4.3, we examine five pivotal terminal conditions at each time step, which are integral spacecraft orbital elements: eccentricity (e), semi-major axis (a_{sm}), inclination (i), right ascension of the ascending node (Ω), and argument of periapsis (ω). These orbital elements are derived from the state (he) elements, as defined in Eq. 1. The conversion of these state elements to orbital elements for convergence assessment proves advantageous, particularly in

Scenarios	Position	State Parameters				
		h (km^2/s)	h_x (km^2/s)	h_y (km^2/s)	e_x	e_y
GTO-1 - GEO	Initial	67288.41965	0	-32107.258	0.7306	0
	Target	29640.2292	0	0	0	0
GTO-2 - GEO	Initial	67246.21689	0	-30529.143	0.7310	0
	Target	29640.2292	0	0	0	0
GTO-3 - GEO	Initial	68196.3519	0	-8311.044	0.725	0
	Target	29640.2292	0	0	0	0
Super-GTO - GEO	Initial	70532.88179	0	-26991.765	0.8705	0
	Target	29640.2292	0	0	0	0
Super-GTO-2 - NRHO	Initial	101055.564	8993.124	-44988.209	0.6342	0.1422
	Target	384073.943	43240.356	-111563.294	-0.0209	-0.1249

Table 2: State parameters (he) defined for all transfer scenarios

GEO transfer scenarios where only three orbital elements (eccentricity (e), a_{sm}, and inclination (i), as detailed in Table 2 of the main text) are required for convergence. This transformation from five to three state elements significantly mitigates the non-linearity of the problem, enhancing convergence and optimizing results. This transformation is also noteworthy in NRHO transfer scenarios, where the convergence of these three orbital parameters substantially influences the remaining two orbital parameters (RAAN, argp) as well.

The computation for converting he elements to orbital state elements is delineated as follows:

The first condition checks the eccentricity of the spacecraft's orbit. The magnitude of eccentricity is calculated from the e_x and e_y parameters, which are part of the state vector. Here e_{tol} denotes the tolerance value for eccentricity.

$$e_{tar} \leq \left[\mathbf{e} = \sqrt{e_x^2 + e_y^2}\right] \leq e_{tar} + e_{tol} \tag{2}$$

The second condition checks the semi-major axis of the spacecraft's orbit, as shown in Eq. (**??**). The semi-major axis (a_{sm}) is calculated using the h, e_x, and e_y parameters from the state vector, as well as the gravitational parameter μ. The value of $\text{a}_{\text{sm}}^{\text{tar}}$ is the desired target value for the semi-major axis, and $\text{a}_{\text{sm}}^{\text{tol}}$ is the tolerance value for the semi-major axis.

$$\text{a}_{\text{sm}}^{\text{tar}} - \text{a}_{\text{sm}}^{\text{tol}} \leq \left[\text{a}_{\text{sm}} = \frac{h^2}{\mu(1 - \sqrt{(e_x^2 - e_y^2)})}\right] \leq \text{a}_{\text{sm}}^{\text{tar}} + \text{a}_{\text{sm}}^{\text{tol}} \tag{3}$$

The third condition checks the inclination angle of the spacecraft's orbit, as shown in Eq. (**??**), where i_{tol} denotes the tolerance value for the inclination angle.

$$i_{tar} \leq \left[i = \sqrt{\frac{h_x^2 + h_y^2}{h}}\right] \leq i_{tar} + i_{tol} \tag{4}$$

The fourth condition assesses the right ascension of the ascending node (RAAN), denoted as (Ω). The calculation involves first determining the vertical component h_z, which is then utilized to find the value of Ω. The computation steps for finding Ω and establishing the tolerance ranges, Ω_{tol}, are detailed as follows:

$$hz = \sqrt{h^2 - h_x^2 - h_y^2} \quad n = cross\left(\begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}h_x\\h_y\\h_z\end{bmatrix}\right) \tag{5}$$

$$\Omega = \begin{cases} \arccos\left(\frac{n(1)}{\|\mathbf{n}\|}\right), & \text{if } n(2) \geq 0 \\ 360 - \arccos\left(\frac{n(1)}{\|\mathbf{n}\|}\right), & \text{otherwise} \end{cases} \tag{6}$$

$$\Omega_{tar} - \Omega_{tol} \leq \Omega \leq \Omega_{tar} \tag{7}$$

The fifth condition evaluates the argument of periapsis (ω). Unlike other parameters, this parameter cannot be directly computed from the he elements. Instead, additional calculations are required to first calculate the rotation matrix and then it determine the value of (ω). The subsequent calculations, along with the corresponding tolerance settings, are outlined as follows:

$$\zeta = \arctan\left(\frac{h_x}{h_z}\right); \quad \eta = -\frac{h_y}{h}; \quad \eta_{\cos} = \sqrt{\frac{h^2 - h_y^2}{h^2}}; \tag{8}$$

$$R_\zeta = \begin{bmatrix} \cos(\zeta) & 0 & -\sin(\zeta) \\ 0 & 1 & 0 \\ \sin(\zeta) & 0 & \cos(\zeta) \end{bmatrix}; \quad R_\eta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \eta_{\cos} & \eta \\ 0 & -\eta & \eta_{\cos} \end{bmatrix}; \tag{9}$$

$$R_{\text{ECI to R}} = R_\eta \cdot R_\zeta; \quad \mathbf{e}_{ECI} = \mathbf{R}^T_{ECItoR} \begin{bmatrix} e_x \\ e_y \\ 0 \end{bmatrix}; \tag{10}$$

$$\omega = \begin{cases} \arccos\left(\frac{\mathbf{n}\cdot\mathbf{e}_{ECI}}{\|\mathbf{n}\|\cdot\|\mathbf{e}_{ECI}\|}\right), & \text{if } e_{ECI}(3) \geq 0 \\ 360 - \arccos\left(\frac{\mathbf{n}\cdot\mathbf{e}_{ECI}}{\|\mathbf{n}\|\cdot\|\mathbf{e}_{ECI}\|}\right), & \text{otherwise} \end{cases} \tag{11}$$

$$\omega_{tar} \leq \omega \leq \omega_{tar} + \omega_{tol} \tag{12}$$

The tolerance ranges and target values for all of these parameters in Eq.(2,3,4,7,12) are presented in Table 2 in the main text.

2.3 Reward Weights

We introduced the gradient-aided reward function in our approach, elaborated in Section 4.3 of the main text. The reward function incorporates user-defined weights, which are presented in Table 3. These weights are represented by $W = [w_1; w_2; w_3]$, where $W[:, 1]$ corresponds to the weightage of a_{sm}, $W[:, 2]$ pertains to the weightage of e, and $W[:, 3]$ reflects the weightage of i in Eq. 16 (main text). These weight assignments are aligned with the first, second, and third columns of Table 3, respectively. We also fixed the value of τ in Eq. 16 (main text) as 0.5. The rewards weights are shown as follows:

	GTO to GEO			**Super-GTO to GEO**			**Super-GTO-2 to NRHO**		
	$W[:,1]$	$\mathbf{W}[:,2]$	$\mathbf{W}[:,3]$	$W[:,1]$	$W[:,2]$	$\mathbf{W}[:,3]$	$W[:,1]$	$W[:,2]$	$\mathbf{W}[:,3]$
$\mathbf{w}_1$	$1e^3$	$2e^3$	$3e^2$	$1e^3$	$4e^3$	$3e^2$	$8e^5$	$8e^5$	$8e^5$
$\mathbf{w}_2$	$1e^{-2}$	$1.9e^{-7}$	$3e^{-5}$	$1e^{-2}$	$1.9e^{-9}$	$3e^{-5}$	$3e^1$	$3e^1$	$3e^1$
$\mathbf{w}_3$	$5e^2$	$7e^2$	$3e^2$	$5e^2$	$2e^3$	$3e^2$	$1e^2$	$1e^2$	$1e^2$

Table 3: Weights (W$= [w_1; w_2; w_3]$) used in calculating the reward function.

3 Implementation Details

In this section, we present the dynamic model assumptions in Section 3.1, eclipse model assumptions in Section 3.2, and hyperparameter settings in Section 3.3. The information is intended to offer comprehensive details to facilitate result replication by any interested party.

3.1 Modeling Assumptions

We assume constant spacecraft thrust, excluding periods in Earth's shadow. Modeling assumptions for the spacecraft's dynamic model include neglecting orbital perturbations and radiation damage. Specifically, we focus on onboard thrust as the sole force, ignoring additional perturbations and assuming constant thrust during the Sun-lit trajectory. These simplifications enable a fair comparison with existing sequential and DRL approaches in the literature. Incorporating orbital perturbations or radiation damage can be done by adding corresponding terms to the model.

In our modeling approach, we maintain the assumption of constant spacecraft thrust, with exceptions only during the spacecraft's passage through the Earth's shadow. Our dynamic model incorporates certain assumptions to streamline the analysis. Firstly, we neglect the influence of orbital perturbations in this paper. While the force terms in Eq 8 (main text) can encompass various forces acting on the spacecraft (thrust, J2 perturbation, gravitational forces) we specifically consider the force to be solely due to onboard thrust. Secondly, we disregard the impact of radiation damage in this paper. In actuality, the spacecraft's solar arrays may experience degradation when traversing the Van Allen belts, leading to a reduction in available thrust. However, we simplify our model by assuming constant thrust during the Sun-lit portion of the trajectory, overlooking the radiation damage effects. These modeling assumptions are made for the purpose of facilitating a fair comparison with the sequential approach [2] and a previously studied Deep Reinforcement Learning (DRL) approach [1] found in the literature. It is important to note that if one wishes to incorporate orbital perturbations into the problem, additive terms representing those perturbations need to be included in Eq 8 (main text). Similarly, for those interested in considering the effect of radiation damage, an artificial neural network-based radiation damage prediction can be integrated into the framework.

3.2 Eclipse Model Assumptions

As the spacecraft undergoes multiple revolutions around the Earth to reach its final orbit employing all-electric propulsion, it is highly likely to traverse the Earth's shadow. During these shadow passages, the spacecraft has the option to utilize onboard batteries to power the thrusters or switch them off and coast. This study assumes coasting during the spacecraft's passage through the Earth's shadow in GEO transfer scenarios.

To identify the regions where the spacecraft enters the Earth's shadow, a shadow model is required. In this work, we employ the cylindrical eclipse model. The cylindrical Earth shadow model assumes that the shadow cast by Earth is cylindrical in shape and remains fixed in space without movement. The conditions to determine whether the spacecraft is in eclipse are defined as follows:

$$X_I < 0, \tag{13}$$

$$\sqrt{Y_I^2 + Z_I^2} < R_E \tag{14}$$

where X_I, Y_I, and Z_I represent the components of the Cartesian position vector of the spacecraft in the Inertial frame, and R_E is the radius of the Earth. The equations to convert the spacecraft's state vector, as utilized in this work, to Cartesian coordinates are discussed in [2].

Learning rate	$3 \exp -4$
Discount factor	0.99
Buffer size	$1 \exp 6$
Time Penalty τ	0.5

Table 4: Hyper-parameters settings used in CDRL Training.

3.3 CDRL Parameters settings

We conducted experiments on an Intel(R) Xeon(R) CPU E5-1620 v4 operating at a frequency of 3.5GHz with 8 cores, coupled with the NVIDIA GeForce GTX 1080 graphics processing unit (GPU), and 32GB of random access memory (RAM) to meet the computational requirements. The implementation of our Cascaded Deep Reinforcement Learning (DRL) algorithms was carried out in Python 3.7 using the PyTorch framework, and built up over the Soft Actor-Critic implementation by stable-baselines. This hardware/software configuration significantly contributed to the efficient and effective development and training of our models. The hyperparameters for our actor, critic, and target critic models are presented in Table 5

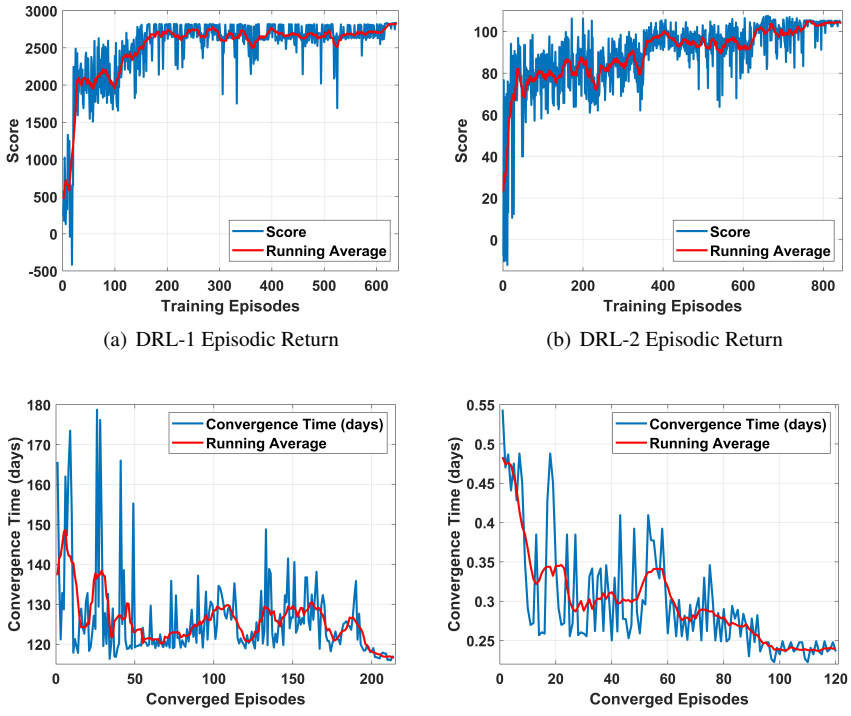
The actor network outputs the mean (μ) and variance (σ) for each action, with the state values as input. Utilizing these mean and variance values, the actor network generates a Gaussian distribution and samples actions from it. Additionally, it produces the log probabilities of the sample distribution to calculate the entropy. The other hyper-parameters utilized in the CDRL training are presented in table 4

Layers	Actor		Critic		Target Critic	
	Size	Activation Fun.	Size	Activation Fun.	Size	Activation Fun.
Input layer	6	ReLU	8	ReLU	8	ReLU
Hidden 1	256	ReLU	256	ReLU	256	ReLU
Hidden 2	256	ReLU	256	ReLU	256	ReLU
output	2	ReLU	1	Linear	1	Linear

Table 5: Network parameters used in CDRL.

4 Additional Training Results

In this section, we present the convergence scores and converging time plots during the training of CDRL networks for two-body scenarios. It's important to note that in the main text, we exclusively showcased these convergence plots for three-body scenarios due to space limitations. As illustrated in Fig. 1 and Fig. 2, it is evident that as the training score increases, the episodic time for the converged episodes decreases. We continued the training of DRL networks until we achieved convergence in episodes along with stable high scores.



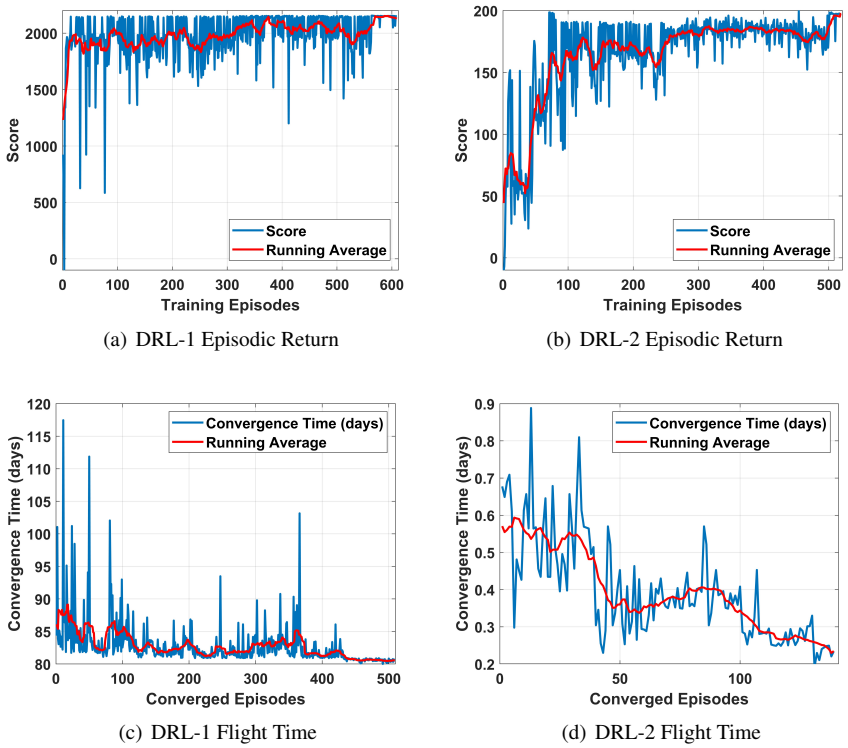
(a) DRL-1 Episodic Return

(b) DRL-2 Episodic Return

(c) DRL-1 Flight Time

(d) DRL-2 Flight Time

Figure 1: Episodic return and convergence times over training steps for GTO to GEO transfer scenario. DRL-1 and DRL-2 represent the two cascaded DRL agents. Episodic return is for all training episodes, whereas the flight time is for converged episodes during the training.



(a) DRL-1 Episodic Return

(b) DRL-2 Episodic Return

(c) DRL-1 Flight Time

(d) DRL-2 Flight Time

Figure 2: Episodic return and convergence times over training steps for Super-GTO to GEO transfer scenario. DRL-1 and DRL-2 represent the two cascaded DRL agents. Episodic return is for all training episodes, whereas the flight time is for converged episodes during the training.

References

[1] Hyeokjoon Kwon, Snyoll Oghim, and Hyochoong Bang. Autonomous guidance for multi-revolution lowthrust orbit transfer via reinforcement learning. AAS 21, 315, 2021.

[2] Suwat Sreesawet and Atri Dutta. Fast and robust computation of low-thrust orbit-raising trajectories. Journal of Guidance, Control, and Dynamics, 41(9):1888–1905, 2018.