AAS 21-302

A HYBRID CLOSED-LOOP GUIDANCE STRATEGY FOR LOW-THRUST SPACECRAFT ENABLED BY NEURAL NETWORKS

Nicholas B. LaFarge[*], Kathleen C. Howell[†], and Richard Linares[‡]

Recent advancements demonstrate machine learning as a potentially effective approach for onboard guidance. Neural network controllers overcome challenging dynamical regions of space and, in contrast to traditional iterative approaches, evaluate in constant time. However, neural networks frequently behave unpredictably, and the resulting control solution is likely to violate practical limits. This investigation proposes a hybrid guidance approach that simultaneously benefits from the speed of neural networks and the robustness of traditional iterative methods to ensure all mission criteria are met. In this paradigm, the neural network rapidly produces accurate startup solutions that undergo several pre-processing techniques, ultimately improving the performance for onboard targeting.

Advancements in onboard autonomy are enabling new opportunities for establishing a sustained human and robotic presence in deep space. In complex multi-body dynamical environments, such as in the Earth-Moon neighborhood, onboard applications for low-thrust spacecraft are particularly challenging. Current onboard guidance techniques are typically iterative, and require sufficiently accurate startup solutions. This investigation demonstrates a neutral network's ability to rapidly provide initial guesses for iterative guidance techniques, and illustrates several pre-processing methods that transform a neural network-computed control history into a suitable startup solution, creating a robust hybrid approach to onboard closed-loop guidance.

Onboard guidance involves mapping state estimates to thrust commands. This mapping is frequently accomplished by applying iterative approaches or optimal control theory. However, several recent investigations employ neural networks to approximate the complex nonlinear control function. Constant evaluation time makes neural networks an attractive tool for onboard use. Recent investigations demonstrate neural network controller efficacy in problems that include planetary landing,[1] asteroid proximity operations,[2] missed thrust analysis,[3] and multi-body guidance.[4] While effective in these problems, one notable limitation in directly employing a neural network as a controller is that the corresponding solution may violate practical constraints of which the network is not aware. For example, prior work frequently assumes that a low-thrust engine possesses no lower bound on thrust, and is able to continuously and instantaneously alter both thrust direction and magnitude over the entire trajectory. The neural network is trained prior to onboard deployment and, hence, is unable to adjust a suggested control history in real time. This investigation addresses this limitation by proposing a hybrid guidance model in which, rather than replacing existing methodology, the neural network augments and improves the initial guess generation process for a traditional iterative approach. In combining neural network and targeting paradigms, the resulting hybrid guidance framework simultaneously benefits from the speed of neural networks and the robustness of targeting to ensure all mission criteria are met.

Trajectory targeting and optimization methods, in general, require suitably accurate startup solutions. In multi-body problems, trajectory designers often generate low-cost initial guesses for transfers by leveraging

[*]Ph.D. Student, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN 47907, USA; nlafarge@purdue.edu
[†]Hsu Lo Distinguished Professor of Aeronautics and Astronautics, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN 47907, USA; howell@purdue.edu
[‡]Charles Stark Draper Assistant Professor, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02138, USA; linaresr@mit.edu

dynamical systems theory and invariant manifolds.[5,6] Dynamical systems-based approaches are leveraged in many previous applications and, when combined with differential corrections and/or optimization techniques, yield optimal solutions for many applications. However, this approach can be computationally intensive and often requires human-in-the-loop interactions. Alternatively, global optimization techniques, such as basin-hopping and evolutionary algorithms, reduce the need for accurate startup solutions,[7] but the corresponding computational complexity renders them infeasible for onboard use. Without a trajectory designer in-the-loop, onboard startup arcs are typically generated in one of two ways. The most common approach is to store a database of optimal solutions on the flight computer, and adjust the solutions as needed during flight.[8] Alternatively, a recent proposed approach is to generate infeasible startup arcs based on two-body approximations, and correct discontinuities with onboard targeting.[9] In multi-body regimes, conic approximations are frequently insufficient, and limit the onboard guidance capability.[6] Neural networks alleviate the need for many pre-compute optimal control histories to be stored in the database approach by computing feasible thrust commands in real-time.

While a similar hybrid guidance paradigm is suggested in previous work,[4] this investigation differs in several notable ways. First, the prior method demands a human in-the-loop to determine which control segments to combine, and does not offer a consistent set of parameters that achieve convergence over a broad range of scenarios. Furthermore, the previous thrust combination technique does not accommodate practical engine limitations, and performs inconsistently in cases where segments are suggested with large variations in thrust direction and/or small thrust magnitude values. Finally, this investigation incorporates reference motion in the initial guess generation process, reducing the computational footprint of the neural network guidance algorithm, and improving convergence consistency.

PROBLEM FORMULATION

The proposed hybrid guidance approach, using neural network and targeting techniques, is evaluated in the Earth-Moon neighborhood. In this area, the three-body problem serves as a suitable dynamical model that is representative of observed natural motion in cislunar space. In particular, the planar Circular Restricted Three-Body Problem (CR3BP) is a useful environment for preliminary evaluation because it both represents a challenging region of space that is relevant to upcoming missions while sufficiently low-fidelity for initial analysis of the hybrid guidance scheme. Additionally, low-thrust propulsion is included to demonstrate algorithmic performance despite limited control authority and pronounced nonlinearities.

Neural Networks

Neural Networks (NNs) are of particular recent interest for potential onboard activities. A neural network is a class of nonlinear statistical models that are frequently employed in machine learning classification and regression tasks.[10] The capability to implement a neural network on a flight computer is under active development, and new hardware technologies will render machine learning approaches more accessible and productive for onboard use.[4] Various methodologies exist for training a neural network controller. Recent investigations focus on leveraging supervised learning and/or reinforcement learning. In a supervised learning approach, many nearby optimal solutions are generated using traditional optimization methods, and a neural network is constructed to approximate the optimal behavior. Alternatively, reinforcement learning produces a neural network controller via direct interaction with the dynamical environment, and does not assume *a-priori* knowledge of the decision making process. Both approaches produce a similar result: a neural network controller that maps state estimates to thrust commands. The proposed hybrid guidance approach employs the end result of training and is, therefore, equally applicable to both methodologies, though specific details may vary depending on the problem definition and network input/output variables.

Several notable investigations leverage supervised learning techniques for spacecraft guidance. Dachwald applied a shallow NN to low-thrust trajectory optimization as early as 2004.[11] More recent investigations in multi-body dynamical regimes involve heteroclinic transfer identification,[12] low-thrust trajectory corrections,[13] and missed thrust analysis.[3] Supervised learning techniques can produce desirable outcomes but carry several notable drawbacks. First, such approaches assume knowledge of the decision-making process. By selecting desired outcomes, the user assumes *a-priori* knowledge of the basis for decision making. This assumption implies that 1) the user-generated data accurately reflects the desired outcomes; and 2) techniques are available to solve the current problem and to generate the data. In regimes where such knowledge is absent, supervised learning techniques are not applicable.

Reinforcement learning techniques are of particular recent interest in spaceflight problems. Reinforcement learning-enabled onboard guidance is broadly categorized based on the phase of flight. Notably productive areas of reinforcement learning research are landing problems[1,14] and small body operations.[2,15] Other investigations include rendezvous,[16] exo-atmospheric interception,[17] stationkeeping,[18] and detection avoidance.[19] Studies that involve low-thrust spacecraft in a multi-body dynamical regime include transfer design using Q-Learning[20] and Proximal Policy Optimization (PPO),[4,21] as well as orbit approach guidance.[22] The neural network controller employed in this investigation is trained using reinforcement learning methodology as detailed by LaFarge et al.[4]

Dynamical Model

The planar CR3BP is a model for the motion of an infinitesimal mass moving under the influence of two celestial bodies within the orbital plane. In this model, two spherically symmetric gravitational bodies, P_1 and P_2 form the primary system as they move in circular orbits about their common barycenter, B; P_3 moves freely with respect to the barycenter in the same plane as P_1 and P_2, depicted in Figure 1. The relative size of the primaries is represented by the mass ratio $\mu = M_2/(M_1 + M_2)$, with M_1 assumed to be the larger of the two bodies. Furthermore, this model assumes that the mass of P_3 is infinitesimal compared to the masses of the primary bodies and, thus, does not influence the motion of the primary system. The position and velocity of P_3, denoted $\boldsymbol{r_3}$, $\boldsymbol{v_3}$, respectively, comprise the vector $\boldsymbol{\rho} = [x \quad y \quad \dot{x} \quad \dot{y}]^T$. The vector components are propagated with respect to the system barycenter, B, in a rotating reference frame, denoted by dashed lines in Figure 1.

Many mission architectures benefit from the inclusion of low-thrust electric propulsion. In contrast to traditional chemical engines, electric propulsive engines are much more efficient, but deliver energy changes over longer time intervals. Low-thrust engines using Solar Electric Propulsion (SEP) include ion thrusters, which are powered through solar panels on the spacecraft. Currently, ion thrusters are successfully employed, for example, on Deep Space 1[23] and Dawn,[24] as well as other missions. Building on this progress, upcoming low-thrust missions include Psyche[25] and the Lunar Gateway.[26] This investigation assumes that P_3 is a spacecraft with a Constant Specific Impulse (CSI) low-thrust engine. The additional propulsion force augments the natural planar CR3BP equations of motion with low thrust terms,

$$\ddot{x} - 2\dot{y} - x = -\frac{(1-\mu)(x+\mu)}{r_{13}^3} - \frac{\mu(x-1+\mu)}{r_{23}^3} + a^{\text{lt}} u_x \tag{1}$$

$$\ddot{y} + 2\dot{x} - y = -\frac{(1-\mu)y}{r_{13}^3} - \frac{\mu y}{r_{23}^3} + a^{\text{lt}} u_y \tag{2}$$

$$\dot{m} = 0 + \frac{-f l^*}{I_{\text{sp}} g_0 t^*} \tag{3}$$

where the right column denotes the additional terms introduced by the low-thrust force, t^* and l^* are the system characteristic time and length, respectively, and $g_0 = 9.80665 \times 10^{-3}$ km/s. The distances between the first and second primary body are defined as $r_{13} = \sqrt{(x+\mu)^2 + y^2}$ and $r_{23} = \sqrt{(x-1+\mu)^2 + y^2}$, respectively. Motion in the CR3BP is nonlinear in a notably sensitive dynamical regime, thus, the proposed guidance strategy exploits the impact of the low-thrust terms to achieve desired behavior. As detailed by Cox et al.,[27] thrust direction is defined by the low-thrust acceleration vector,

$$\boldsymbol{a}^{\text{lt}} = \frac{f}{m}\hat{u} = (a^{\text{lt}} u_x)\hat{x} + (a^{\text{lt}} u_y)\hat{y} \tag{4}$$

where f is the nondimensional thrust magnitude, $m = M_3/M_{3,0}$ is the nondimensional spacecraft mass, and M_3 is the mass of the spacecraft at the beginning of the thrusting segment. The thrust direction is oriented

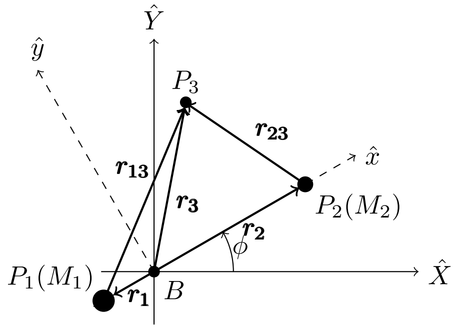


Figure 1. Planar CR3BP vector definitions. The rotating frame $(\hat{x}, \hat{y})$ is oriented with respect an inertial reference frame $(\hat{X}, \hat{Y})$ by an angle ϕ. Modified from Cox.[4,28]

by a unit vector, $\hat{u}$, fixed in the rotating frame. Over any integration segment, thrust is assumed fixed in the CR3BP rotating frame. While a continuously changing inertial thrust direction may not be practical, precession of the rotating frame during thrusting time intervals may be addressed when transferring CR3BP solutions into a higher-fidelity model. Furthermore, propulsive capability is inversely related to spacecraft mass and, hence, as propellant is expended, the spacecraft gains more thrust. The nondimensional thrust magnitude is computed as, $f = \frac{Ft^*}{l^* M_{3,0}}$, where F is thrust in kilonewtons and $M_{3,0}$ is the initial mass of the spacecraft in kilograms. Thrust magnitude is also expressed in terms of throttle $T \in [0, 1]$, where $f = T f_{\text{max}}$ relates the throttle value and the nondimensional thrust magnitude..

This investigation includes a sample spacecraft with maximum propulsive capability of $f_{\text{max}} = 0.04$. A comparison between this sample spacecraft and other previous and planned engine capabilities is summarized in LaFarge et al., Table 1.[4] The sample spacecraft possesses similar propulsive capability to the planned Psyche spacecraft, which is greater than Hayabusa 1, Hayabusa 2, Dawn, and Lunar IceCube, but less than Deep Space 1. As a nondimensional quantity, this thrust level represents any spacecraft with the same thrust-to-mass ratio. For example, $f_{\text{max}} = 0.04$ models an 11.46 kg spacecraft with a BIT-3 CubeSat engine[29] (planned for use on Lunar IceCube and LunaH-Map) as well as an 850.2 kg spacecraft with an NSTAR engine[23] (used on Deep Space 1 and Dawn). Low thrust engines also commonly possess a lower limit on thrust level, f_{engine}. This investigation assumes a lower throttle limit of 58%, $f_{\text{engine}} = 0.58 f_{\text{max}}$, which is consistent with the observed limitations on Hayabusa 1.[30]

Omitting low-thrust terms, the natural planar CR3BP equations of motion yield a well-known integral, i.e., the Jacobi constant of integration (C). This single integral of motion is evaluated as,

$$C = 2\left(\frac{1-\mu}{r_{13}} + \frac{\mu}{r_{23}} + \frac{1}{2}\left(x^2 + y^2\right)\right) - \left(\dot{x}^2 + \dot{y}^2\right) \tag{5}$$

The Jacobi constant is related to orbital energy as observed in the rotating frame, and remains constant throughout any natural propagation. This integral offers key insight into the limiting bounds for possible motion of P_3 and supplies a useful scalar metric for deviations relative to a reference trajectory where C is preserved. Without a known analytical solution in the CR3BP, numerical integration methods are leveraged to produce a time history stemming from an initial value problem. For implementation, it is useful to express the equations of motion, in Equations (1)–(3), as five coupled first-order equations that are then solved using numerical integration techniques. For this investigation, a Runge-Kutta-Fehlberg 78 integration scheme is employed.

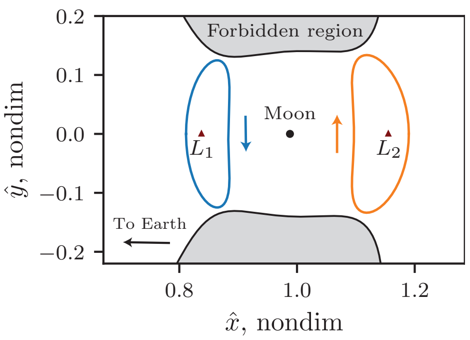
Scenario Overview

The proposed hybrid guidance approach is evaluated using the mission application introduced previously in LaFarge et al.[4] In this application, the Earth-Moon CR3BP is employed due to recent interest in cislunar space, with characteristic quantities and low-thrust CSI engine parameters listed in Table 1. The sample scenario is characterized by a planar, heteroclinic transfer between L_1 and L_2 Lyapunov orbits at a Jacobi constant value of $C = 3.124102$, illustrated in Figure 2 where Figure 2(a) depicts the periodic orbits, and Figure 2(b) illustrates a continuous heteroclinic transfer from the L_1 Lyapunov orbit to the L_2 Lyapunov orbit. In the given mission application, a guidance scheme is tasked with recovering from a deviation relative to the reference trajectory in Figure 2(b). A neural network is trained to recover from initial deviations modeled as $3\sigma_r = 1000$ km and $3\sigma_v = 10$ m/s. Orbit Determination (OD) navigation errors on the order of $3\sigma = 1$ km and 1 cm/s[31] provide a useful comparison metric for the perturbation distribution. During training, initial deviations are three orders of magnitude larger than expected OD disturbances. Orbit determination errors are included as a comparison metric for the initial perturbation distributions, and are not implemented during simulations.

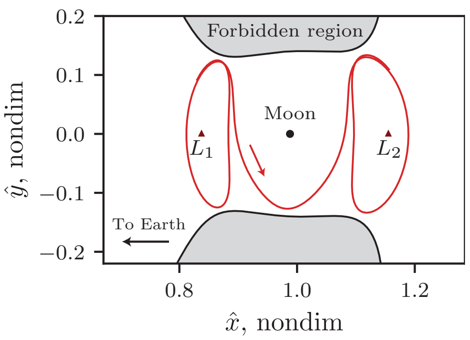
In prior work,[4] the neural network controller recovers from introduced deviations, completes a given transfer, and reaches a pre-defined arrival criteria in more than 99% of sample trials when $3\sigma_r = 1000$ km and $3\sigma_v = 10$ m/s. While remarkably consistent, there are several notable limitations observed in the corresponding solutions. First, the neural network is unable to produce coasting segments (where $f = 0$). To illustrate this phenomenon, a sample NN trajectory and control history is plotted in Figure 3. In this scenario, the neural network recovers from a 1108 km, 6.7 m/s initial deviation, and successfully completes the given transfer. However, despite satisfying the stated goal of returning to the reference, the majority of the NN-generated transfer involves oscillations at very low levels of thrust, depicted in Figure 3(b). Furthermore, the position and velocity relative to the reference trajectory for the scenario in Figure 3(a), as a normal correspondence, is plotted in Figure 3(c). While the preliminary deviations are large, the initial thrusting segments decrease the relative distances until both position and velocity errors remain bounded. These errors result from small inaccuracies in the neural network solution, causing a constant error introduction/reduction cycle. The neural network solution is most effective for the initial segments with higher levels of thrust, and less accurate for later segments with lower thrust values.

Table 1. Characteristic quantities and low-thrust CSI engine parameters.[4]

Earth-Moon Characteristic Quantities		Low-Thrust Engine Properties	
μ, nondim	0.012004715741012	f_{max}, nondim	0.04
l^*, km	384747.962856037	I_{sp}, s	3000
t^*, s	375727.551633535	f_{engine}	$0.58 f_{\text{max}}$

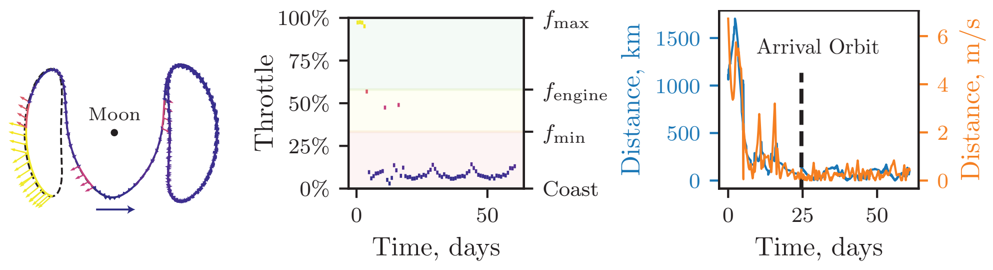


(a) L_1 and L_2 Lyapunov orbits.



(b) Heteroclinic transfer.

Figure 2. Periodic orbits and heteroclinic transfer in the Earth-Moon system applied to the sample scenario. All motion at $C = 3.124102$, with the corresponding forbidden regions in gray. Modified from LaFarge et al., Figure 7.[4]



(a) Sample NN trajectory (b) Thrust magnitude history. (c) Variation from reference trajectory.

Figure 3. Sample neural network-driven solution simulated for 61 days. Figures (a), (b) modified from LaFarge et al., Figure 11.[4]

The oscillatory behavior observed at lower thrust levels in Figure 3(b) is likely unnecessary for solving the given problem and is, therefore, not included in the hybrid guidance process. Furthermore, when f is small, sensitivity to thrust direction is greatly decreased, causing less useful information contained within specific $\hat{u}$ values. A threshold, f_{min}, is heuristically selected to ensure the oscillatory behavior is discarded, signified by the color red in Figure 3(b). The value for f_{min} is specific to the observed neural network behavior. For this investigation, $f_{\text{min}} = 0.33 f_{\text{max}}$ provides a useful lower bound. Above f_{min}, thrust values are separated depending on their physical viability for the given engine. Visualized in Figure 3(b), yellow indicates that the given thrust value falls below the practical limit, f_{engine}, while green reflects the fact that the given thrust magnitude is realistic.

CONTROL COMBINATION AND ADJUSTMENT

Several techniques are employed to combine and scale successive thrust segments. Neural network-produced guidance solutions often involve many sequential control arcs. In practice, however, a single thrust command is often sufficient for recovering a low thrust spacecraft from deviations. This investigation demonstrates an approach for creating an initial guess for a single unified thrust segment given multiple discrete arcs, and offers a method for transforming thrust magnitude into practical bounds for low-thrust engines.

Thrust Segment Combination

Control values are assumed constant over integration segments and, hence, subsequent thrust arcs include an assumed instantaneous attitude and power shift as the spacecraft transitions from one segment to another. The inclusion of multiple sequential thrust segments is, for many applications, an artifact of the employed numerical method that is implemented and not always necessary for solving the given problem. In these cases, a strategy for combining multiple control arcs is useful to yield a single segment that is representative of the combined effect of each discrete component.

Combining subsequent control segments is particularly productive for neural network controllers that do not estimate time in the output layer and, instead, assume all control values offered by the neural network occur over a pre-defined time interval. For reinforcement learning approaches, in particular, these time intervals are typically short and, hence, a solution may include many interactions with the neural network controller. In these cases, computing a unified guess for thrust magnitude, direction, and time is challenging due to occasional large shifts in control values between successive thrust arcs produced by the neural network. To address these discontinuities, variation in thrust direction and mass are incorporated into the combination process to yield accurate initial guesses that are representative of the neural network-suggested motion. Furthermore, smoothing of the discontinuities may be incorporated into the neural network training process by, for example, penalizing a change in thrust direction,[22] but this possibility is not currently investigated.

The representative thrust vector across the multiple control segments is termed the 'combined' thrust arc. The direction of this combined vector is defined as the unit vector directed along the average low thrust acceleration vector as defined in Equation (4). Assuming n segments are included, the vector $\hat{u}_{\text{avg}}$ is evaluated as,

$$\hat{u}_{\text{avg}} = \frac{\boldsymbol{a}^{\text{lt}}_{\text{avg}}}{|\boldsymbol{a}^{\text{lt}}_{\text{avg}}|} \quad \text{where} \quad \boldsymbol{a}^{\text{lt}}_{\text{avg}} = \frac{\sum_{i=1}^{n} \boldsymbol{a}^{\text{lt}}_{i}}{n} \tag{6}$$

An alternative approach directly averages the thrust directions of the individual segments, $\hat{u}_i$, which does not incorporate the thrust level along each path. With the goal of creating a single representative segment, determining the control direction via the acceleration vectors via Equation (6) reflects a belief that segments with higher thrust values impose larger impacts on the unified solution. Then, the combined nondimensional thrust magnitude, f_{avg}, is derived from multiplying the acceleration vector magnitude by the initial nondimensional mass value, i.e.,

$$f_{\text{avg}} = m_0 |\boldsymbol{a}^{\text{lt}}_{\text{avg}}| \tag{7}$$

The expressions for $\hat{u}_{\text{avg}}$ and f_{avg} assume the feasibility of representing the discrete thrust segments by a single thrust arc. As propagation time and control complexity increases, the set of n control arcs may be subdivided into multiple subsets to produce more accurate estimations.

A key challenge in combining thrust segments is estimating the final propagation time. If n segments each possess a time Δt, the total sum $n\Delta t$ frequently results in overestimations that prohibit convergence. To combat this phenomenon, a weighted sum for total thrust time, τ^-, is computed as,

$$\tau^- = \left| \sum_{i=1}^{n} \left(\frac{f_i}{f_{\text{max}}} \right) \left(\frac{m_0}{m_i} \right) u_i \Delta t \right| \tag{8}$$

where f_i/f_{max} prioritizes segments with larger thrust magnitudes such that they carry greater influence in the propagation time summation; m_0/m_i then compensates for mass variation across the segments. Including $\hat{u}_i$ in the summation decreases the thrust time estimate when large variations in control occur. Rearranging Equation (8), and substituting the expressions in Equations (6) and (7), yields,

$$\tau^- = \frac{m_0 \Delta t}{f_{\text{max}}} \left| \sum_{i=1}^{n} \left(\frac{f_i}{m_i} \right) \hat{u}_i \right| = \frac{m_0 \Delta t}{f_{\text{max}}} n \left| \boldsymbol{a}^{\text{lt}}_{\text{avg}} \right| = \frac{f_{\text{avg}}}{f_{\text{max}}} n \Delta t \tag{9}$$

The combined thrust segment is defined by the three quantities $\hat{u}_{\text{avg}}$, f_{avg} and τ^-.

Thrust Magnitude Adjustment

Many low-thrust engines are not physically able to produce thrust values below a certain magnitude, f_{engine}. For example, Hayabusa 1 and 2 are only capable of maneuvers above 58% and 70% of maximum thrust (f_{max}), respectively.[30] When suggested thrust segments fall below f_{engine}, it is useful to leverage the relationship between thrust time and magnitude to compute a new guess within feasible bounds. For a CSI engine, the mass flow rate, defined in Equation (3), is a function of nondimensional thrust magnitude f. This equation is decoupled and is, hence, easily integrated to express spacecraft mass as a function of time,

$$m(f,t) = \frac{-fl^*}{I_{\text{sp}} g_0 t^*} t + m_0 \tag{10}$$

If the original spacecraft mass equals m_0 at time $t_0 = 0$ and the thrust is active at f magnitude until t_f, the final mass ratio m_f/m_0 is expressed as,

$$\frac{m_f}{m_0} = \frac{m(f,t_f)}{m(f,t_0)} = \frac{-fl^*}{I_{\text{sp}} g_0 t^* m_0} t_f + 1 \tag{11}$$

Equation (11) provides an expression for the remaining mass after a thrusting segment. Alternatively, rearranging this expression and solving for t_f yields a relationship for thrusting time as a function of propellant consumption,

$$t_f(\Delta m, f) = \frac{-I_{\text{sp}} g_0 t^*}{fl^*} \Delta m \tag{12}$$

where $\Delta m = m_f - m_0$ is the expended propellant. For thrust segments where $f < f_{\text{engine}}$, Equation (12) supplies a relationship to scale both thrust magnitude and time. In this approach, a new thrust magnitude f^+ is selected such that $f^+ \in [f_{\text{engine}}, f_{\text{max}}]$, and Equation (12) yields a new value for thrusting time that keeps propellant consumption constant. An alternative, but equivalent, relationship involves thrusting time as a function of equivalent ΔV. From the ideal rocket equation, change in velocity for a CSI engine is expressed as,

$$\Delta V = I_{\text{sp}} g_0 \log\left(\frac{m_0}{m_f}\right) \tag{13}$$

Rearranging this expression yields the final mass ratio m_f/m_0 as a function of ΔV,

$$\frac{m_f}{m_0} = \frac{m_0}{\exp(\Delta V / I_{\text{sp}} g_0)} \tag{14}$$

Equation (14) is equivalent to Equation (11). Equating the two expressions and solving for t_f yields,

$$t_f(\Delta V, f) = \frac{-I_{\text{sp}} g_0 t^* m_0}{f l^*}\left(\frac{1}{\exp(\Delta V / I_{\text{sp}} g_0)} - 1\right) \tag{15}$$

The derived functions for $t_f(\Delta m, f)$ and $t_f(\Delta V, f)$ offer simple, computationally efficient methods for adjusting thrust magnitudes and thrust times to compute initial guesses that are practical for low-thrust engines. These scaling functions are visualized in Figure 5. Adjusting thrust time to maintain equivalent ΔV and final mass as constants corresponds to placing a horizontal line across the plot at the desired constant level and observing where this value intersects the different throttle percentage lines.

Any thrust magnitude greater than a specified lower bound, f_{engine}, may be arbitrarily selected to transform the segment into practical bounds for a given engine. Optimal control theory is leveraged in this investigation to select the final thrust magnitude, f^+. For CSI engines in a two-body model, Longuski et al. derive a bang-bang optimal control law that alternates between null-thrust (coast) and maximum-thrust (f_{max}) arcs.[32] While this analysis does not involve optimization, $f^+ = f_{\text{max}}$ is selected as the final thrust magnitude to bias the initial guess toward a propellant-efficient solution. The resulting thrusting time is observed as the intersection between the 100% throttle line in Figure 5 and the equivalent ΔV from the ideal rocket equation, Equation (13), and is evaluated as,

$$\tau^+ = \frac{-I_{\text{sp}} g_0 t^* m_0}{f_{\text{max}} l^*}\left(\frac{1}{\exp(\Delta V / I_{\text{sp}} g_0)} - 1\right) \tag{16}$$

The resulting thrust segment defined by $\hat{u}_{\text{avg}}, f_{\text{max}}, \tau^+$ form the complete transformed solution, which may then be employed to construct an initial guess for a traditional iterative guidance approach.

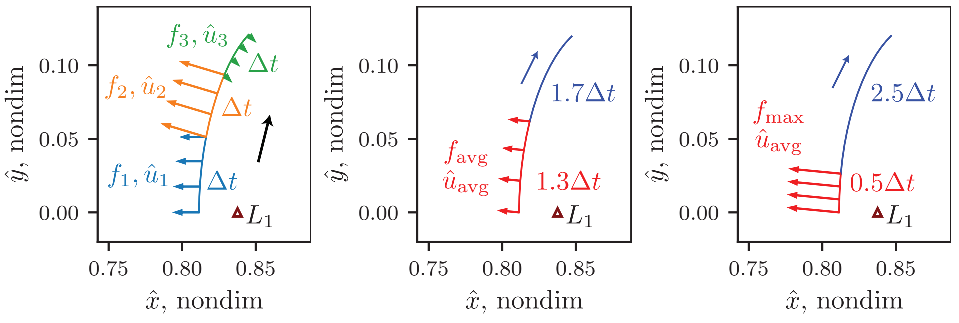
Sample Combination Process

To illustrate the complete combination process, a sample sequence of three thrust arcs in the vicinity of the L_1 Lyapunov point in the Earth-Moon system is depicted in Figure 4(a), where the arrow directions and lengths signify thrust direction and magnitude, respectively, and $\Delta t \approx 20.87$ hours. Specific thrust command values are summarized in Table 2. For this sample scenario, Equations (6), (7) and (9) are leveraged to combine the three sample segments into the single representative thrust arc in Figure 4(b). In this case, including $\hat{u}_i$ in the computation of f_{avg} and τ^- lessens the impact of the third arc on the overall solution since f_3 is only 20% of f_{max}.

While the thrusting segment in Figure 4(b) resembles Figure 4(a), the simulated low-thrust engine is not physically able to produce a propulsive force at this magnitude, since $f_{\text{avg}} < f_{\text{engine}}$. Considering the potential implementation of the initial guess, Equation (15) is leveraged to compute the thrust time that achieves the same equivalent ΔV (4.11 m/s) when the engine thrusts at maximum power, f_{max}. This conversion is visualized in Figure 5, where the thrust level corresponding to Figure 4(b) is red, and maximum thrust is black. Thrust time is scaled between 41% and 100% by noting the intersection points along the dashed line, ensuring that equivalent ΔV and mass consumption are preserved.

Table 2. Thrust commands for sample scenario depicted in Figure 4.

Segment	f	$\hat{u} = [u_x, u_y]$	t, nondim	t, hours	Equiv. ΔV, m/s
1	$0.5 f_{\text{max}}$	$[-1, 0]$	0.2	20.87	4.10
2	$0.9 f_{\text{max}}$	$[-0.9578, 0.2873]$	0.2	20.87	7.38
2	$0.2 f_{\text{max}}$	$[0.7071, -0.7071]$	0.2	20.87	1.64
Combined	$0.41 f_{\text{max}}$	$[-0.9954, 0.0955]$	0.245	25.60	4.11
Adjusted	f_{max}	$[-0.9954, 0.0955]$	0.100	10.46	4.11



(a) Sample segments (b) Combined solution (c) Adjusted solution

Figure 4. Sample thrust arc sequence combination in the Earth-Moon system. The three discrete thrust segments (a) are combined into a single thrust arc where total thrust time is decreased by 56.7% (b). Thrust command values are listed in Table 2.

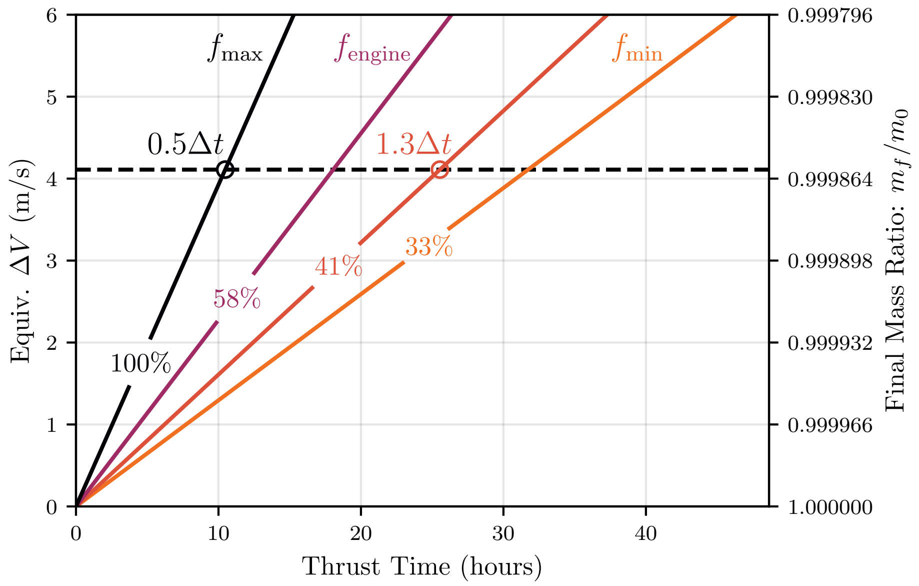


Figure 5. Relationship between propellant consumption, equivalent ΔV and thrust time for an ideal rocket with a CSI engine. Horizontal line denotes the thrust time adjustment for increasing f_{avg} (red) to f_{max} (black) for the segment depicted Figure 4(b), resulting in a new thrust arc in Figure 4(c).

HYBRID GUIDANCE ARCHITECTURE

Guidance, Navigation, and Control (GN&C) comprise three main components for autonomous flight systems. In this paradigm, guidance is tasked with planning a suitable trajectory that satisfies mission criteria. In both pre-flight and ground-based analyses, where computational resources are abundant, path-planning is often formulated as an optimization problem to minimize propellant expenditure. Many approaches exist for low-thrust optimal guidance, including nonlinear programming[33] and global optimization techniques.[7] However, to reduce the computational footprint for in-flight scenarios, alternative onboard approaches prioritize feasibility over optimality.[9] In particular, differential corrections, or *targeting*, is planned for Orion's onboard guidance.[9]

The proposed hybrid guidance framework is subdivided into three discrete components, depicted in Figure 6. First, the "Neural Network Guidance" step involves tasking a trained neural network controller with mapping a given state estimate to a path and control history that returns the spacecraft to the vicinity of a reference trajectory. Next, "Trajectory Pre-Processing" leverages Equations (6), (7) and (16) to transform the neural network solution into a single maneuver within practical engine limits. Finally, a "Differential Corrections" step is employed to construct a continuous, feasible transfer.

Neural Network Guidance

In previous work,[4] a trained neural network is leveraged to suggest many sequential thrusting segments that perform three tasks: 1) return a spacecraft to a given reference trajectory, 2) complete the given transfer, and 3) maintain a target orbit. This process results in many discrete segments, which poses practical challenges for both propellant consumption and implementation. In the proposed hybrid approach, however, the neural network is only tasked with recovering from a particular perturbation, and is not required to complete the full transfer. Simplifying the given task for the neural network controller reduces its computational footprint, and creates fewer opportunities for unexpected results. This investigation demonstrates a blended functionality leveraging both neural network and iterative techniques and, hence, the training process itself is not considered as a focus here.

The NN-driven perturbation recovery process is summarized in Figure 7. The neural network is trained to suggest thrust segments with Δt durations. Beginning with an initial state estimate, assumed to be computed by a navigation system, the neural network determines a thrusting segment with magnitude (f_1) and direction ($\hat{u}_1$). Integrating forward Δt in time yields a new state as an input to the network. This iterative process continues until one of several termination criteria are met. First, the algorithm terminates if the suggested thrust (f_i) or the combined effect of averaging the suggested control segments (f_{avg}) results in a nearly-coasting arc, i.e., $f_i < f_{\text{min}}$ or $f_{\text{avg}} < f_{\text{min}}$, where f_{min} is visualized in Figure 3(b). This criteria assumes that the neural network tends to output low values for f_i when the distance to the reference trajectory is small and, hence, indicates that the recovery task is complete. Alternatively, NN guidance failure occurs when the relative deviation from the reference trajectory exceeds pre-defined thresholds in position or velocity: 8000 km and 35 m/s, respectively. When deviation occurs, NN guidance terminates to avoid further deviation.

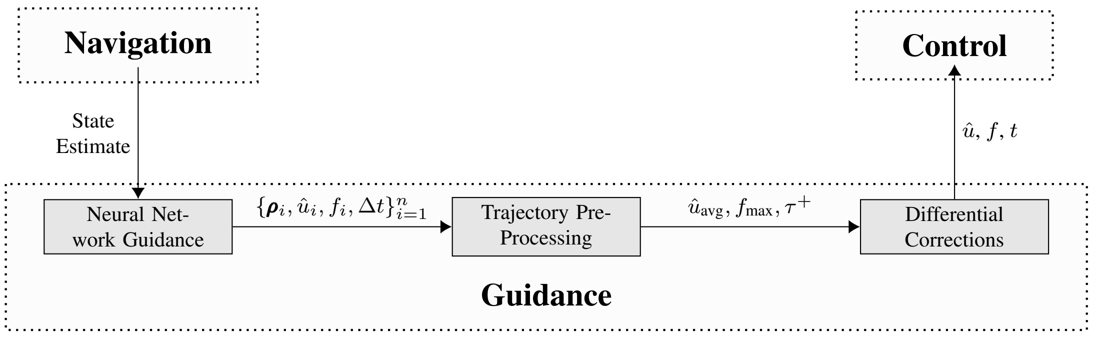

Navigation
Control
State Estimate
$\hat{u}, f, t$
Neural Network Guidance
$\{\boldsymbol{\rho}_i, \hat{u}_i, f_i, \Delta t\}_{i=1}^{n}$
Trajectory Pre-Processing
$\hat{u}_{\text{avg}}, f_{\text{max}}, \tau^+$
Differential Corrections
Guidance

Figure 6. Information flow through hybrid guidance process

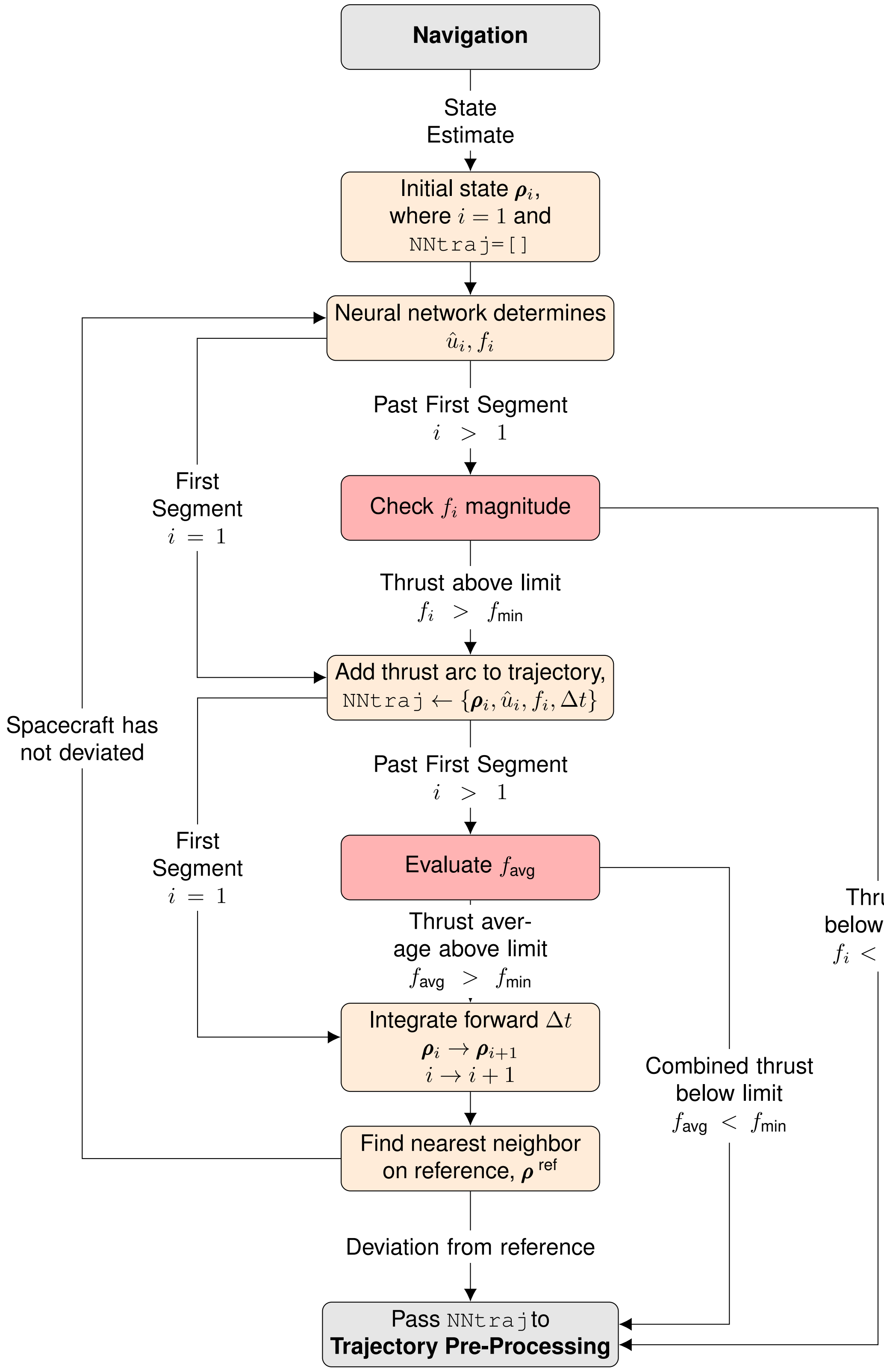

Navigation
State Estimate
Initial state $\boldsymbol{\rho}_i$, where $i = 1$ and `NNtraj=[]`
Neural network determines $\hat{u}_i, f_i$
Past First Segment $i > 1$
Check f_i magnitude
First Segment $i = 1$
Thrust above limit $f_i > f_{\text{min}}$
Add thrust arc to trajectory, `NNtraj` $\leftarrow \{\boldsymbol{\rho}_i, \hat{u}_i, f_i, \Delta t\}$
Spacecraft has not deviated
Past First Segment $i > 1$
Evaluate f_{avg}
First Segment $i = 1$
Thrust below limit $f_i < f_{\text{min}}$
Thrust average above limit $f_{\text{avg}} > f_{\text{min}}$
Integrate forward Δt $\boldsymbol{\rho}_i \rightarrow \boldsymbol{\rho}_{i+1}$ $i \rightarrow i+1$
Combined thrust below limit $f_{\text{avg}} < f_{\text{min}}$
Find nearest neighbor on reference, $\boldsymbol{\rho}^{\text{ref}}$
Deviation from reference
Pass `NNtraj` to **Trajectory Pre-Processing**

Figure 7. Neural network guidance process

Finally, for cases in which the NN delays in its response to a perturbation, i.e., initially produces a nearly-coasting segment followed by a large amount of thrust, at least two thrust arcs are always included along the trajectory. The final output of the neural network guidance step is a trajectory consisting of dynamical states ($\boldsymbol{\rho}_i$), control variables ($\hat{u}_i, f_i$), and integration times Δt, resulting in n discrete arcs,

$$\texttt{NNtraj} = \{\boldsymbol{\rho}_i, \hat{u}_i, f_i, \Delta t\}_{i=1}^{n} \tag{17}$$

where the final segment at $\boldsymbol{\rho}_n$, integrated for Δt with $\hat{u}_n$ and f_n control, is assumed to terminate in the vicinity of the reference trajectory.

Three representative scenarios are depicted in Figures 8–10. For each scenario, plot (a) depicts the throttle control history for NN guidance, where the colored horizontal lines represent NN-generated nondimensional thrust magnitude values (f_i) over time, and gray dashed lines signify the running value for the combined thrust magnitude value (f_{avg}), updated given each new control segment i by Equation (7), where n is the current time step. The complete NN guidance trajectory and control history is visualized in the first plot of (b) for each scenario. The three scenarios are each summarized as follows:

- Scenario 1 (Figure 8) represents the nominal behavior of the neural network. In this case, the network selects three sequential thrust segments without large direction changes, and a fourth thrust arc that possesses a small throttle value. The neural network guidance step recognizes that the spacecraft reaches the vicinity of the reference trajectory by observing $f_4 < f_{\text{min}}$, plotted in Figure 8(a). Crossing the f_{min} threshold terminates the neural network guidance process, and the f_4 is not included in the final solution.
- Scenario 2 (Figure 9) illustrates a case where neural network guidance terminates when the average thrust is small, i.e., $f_{\text{avg}} < f_{\text{min}}$. Due to the large discrepancy between $\hat{u}_1$ and $\hat{u}_2$, visualized in Figure 9(b) (NN Guidance), f_{avg} is characterized by a small throttle magnitude despite both f_1 and f_2 possessing large magnitudes, depicted in Figure 9(a).
- Scenario 3 (Figure 10) demonstrates the case where neural network guidance proceeds past the first segment despite $f_1 < f_{\text{min}}$. This behavior corresponds to logic in Figure 7 that causes the f_i and f_{avg} evaluation steps to be skipped when $i = 1$. In this case, the neural network is delayed in its response to the perturbation, and a large second thrust magnitude, f_2, is computed, depicted in Figure 10(a). This second segment is added to the neural network solution, and guidance terminates when $f_3 < f_{\text{min}}$ occurs.

Trajectory Pre-Processing

Once the NN guidance process is complete, the resulting solution $\{\boldsymbol{\rho}_i, \hat{u}_i, f_i, \Delta t\}_{i=1}^{n}$ is passed to a pre-processing step. This trajectory includes n discrete thrusting arcs, where segment i is comprised of $\boldsymbol{\rho}_i$ (the segment's initial position and velocity), $\hat{u}_i$ (thrust direction defined in the rotating reference frame), f_i (nondimensional thrust magnitude), and Δt (thrust time). The neural network that is incorporated does not include time as an output and, hence, Δt is the same for all segments (20.87 hours for sample cases).

The pre-processing step employs several transformation techniques that generally improve the startup solution for the corrections algorithm. An alternative approach may omit pre-processing and pass the neural network-generated path directly to the corrections algorithm. However, passing over this step leads to solutions with more variations in thrust direction, and may prohibit convergence in cases where the neural network significantly violates a practical constraint, e.g. $f << f_{\text{engine}}$. Two techniques, in particular, are employed during pre-processing. First, the proposed guidance framework assumes deviations may be recovered using a single thrust segment. Hence, when the neural network includes multiple arcs, it is useful to employ weighted averaging techniques to compute a single representative thrust arc, as derived in Equations (6) and (7). Second, to ensure feasibility, thrust magnitude and time are scaled to guarantee that the initial guess falls within practical bounds for the given engine characteristics using the relationship in Equation (16).

The result of pre-processing includes a determination of the necessity for an active thrust sequence for a given state estimate. Identifying coasting segments is challenging when directly employing a NN controller.

In the hybrid guidance model, if the thrust combination process yields a small average magnitude, i.e., $f_{\text{avg}} < f_{\text{min}}$, the system assumes that coasting is appropriate and does not continue on to the corrections process. This investigation assumes 6-hour intervals for coast arcs before the entire hybrid guidance process is repeated at the new state.

Pre-processing the results for the representative sample scenarios are depicted in Figures 8(b), 9(b) and 10(b). Each scenario includes the neural network-computed control segments ("NN Guidance"), the merged solution determined by Equations (6) and (7) ("Combined"), and the final transformed result where $f_{\text{avg}} \rightarrow f_{\text{max}}$, Equation (16) ("Adjusted"). The result of pre-processing each scenario is summarized below:

- Scenario 1: The direction of the combined and adjusted solutions closely resembles the NN-generated arcs, with a 12.9-hour reduction in thrust time for the combined case, and an additional 10.25-hour reduction for the final adjusted solution.
- Scenario 2: A large change in the thrust direction is suggested by the neural network. In this case, the $136°$ difference between $\hat{u}_1$ and $\hat{u}_2$, observed in Figure 9(b), results in a combined solution that drastically reduces the thrust magnitude. With $f_{\text{avg}} < f_{\text{min}}$, the combined solution indicates that coasting is appropriate for the original state estimate, and the scaling step is not necessary.
- Scenario 3: Illustrates the behavior when a large disparity in thrust magnitude exists, in this case, $f_1 << f_2$. The combined control variables are computed by weighting thrust magnitude and, hence, f_2 exerts a larger impact on the resulting solution than f_1. This disparity is observed in the combined thrust direction closely resembling $\hat{u}_2$.

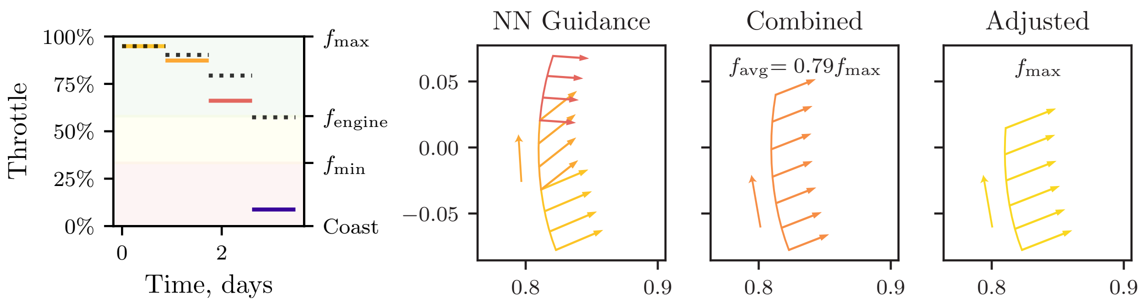
Differential Corrections

A differential corrections, or *targeting*, algorithm is employed as the final step of the hybrid guidance approach to construct a complete initial guess, correct discontinuities, and to ensure mission criteria are met. This investigation implements a direct multiple shooting strategy to produce feasible thrust commands that are passed to the control system. If sufficient computational resources are available, optimization techniques are a potential alternative approach for the targeting step, however this possibility is not investigated.

Targeting is a common numerical technique in astrodynamics for constructing continuous trajectories, and are suitable for different aspects of the low-thrust CR3BP.[13] Targeters are formed as a multidimensional generalization of a Newton-Raphson method.[34] In particular, this investigation employs a multiple shooting strategy to simultaneously accommodate numerous types of arcs and control variables across a trajectory. This targeting approach is implemented by varying an array of $\mathbb{k}$ design variables, $\boldsymbol{X}$, to satisfy a set of $\mathbb{m}$ scalar constraints, $\boldsymbol{F}(\boldsymbol{X}) = \mathbf{0}$. Similar to other iterative approaches, targeting algorithms require sufficiently accurate startup solutions. Selecting an initial guess is challenging in complex dynamical regions of space and, hence, the hybrid guidance approach benefits from the startup arc accuracy and speed offered by a neural network controller.

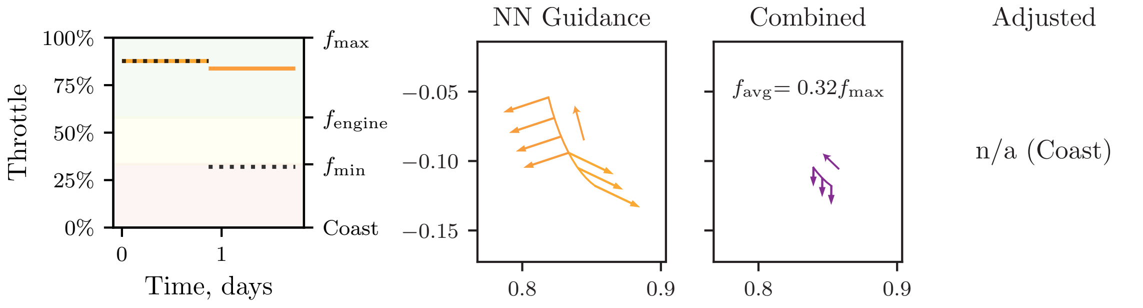
The targeting startup solution is comprised of three parts, depicted in Figure 11. First, the red arc represents the thrust segment produced from the neural network guidance and pre-processing steps. This initial thrust segment originates at a fixed initial state $\boldsymbol{\rho}_0$, and is parameterized by θ_0, f_0, and t_0. Startup values for control variables are received from the pre-processing step, where, initially, θ_0 is the angle between $\hat{u}_{\text{avg}}$ and the rotating $\hat{x}$ axis, $f_0 = f_{\text{max}}$, and $t_0 = \tau^+$. The neural network only affects this initial control segment, with the remainder of the initial guess for the targeting process constructed from the original reference trajectory and target orbit. The second part of the startup solution, plotted in blue, corresponds to the initial reference trajectory depicted in Figure 2(b). To minimize state discontinuity, the initial state along the reference $\boldsymbol{\rho}_1$ is selected by conducting a nearest-neighbor search in position and velocity between the final state along the thrust arc, $\boldsymbol{\rho}_{0,f}$, and the reference trajectory. The reference trajectory time, t_1, is simply the time between $\boldsymbol{\rho}_1$ and the end of the reference trajectory.

Revolutions of the arrival orbit comprise the final step of the startup solution, represented by the black orbit in Figure 11. In this approach, a nearest-neighbor search is again conducted between the end state along the



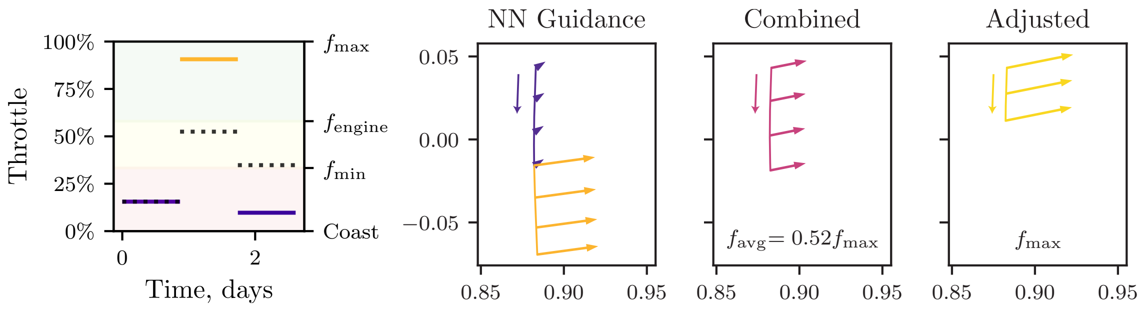
(a) NN guidance thrust magnitudes (b) Pre-Processing Steps ($\hat{x}$-$\hat{y}$,nondim)

Figure 8. Scenario 1: Nominal neural network behavior. Controller recovers from deviation in three steps, ceasing iteration when control values are produced such that $f_4 < f_{\min}$. Total thrust time begins at 62.62 hours, reduces to 49.71 hours when combined, and results in 39.71 hours after adjustment to $f_{\max}$. Solid and dashed lines indicate values for f_i and f_{avg}, respectively.



(a) NN guidance thrust magnitudes (b) Pre-Processing Steps ($\hat{x}$-$\hat{y}$,nondim)

Figure 9. Scenario 2: Behavior when the average low thrust magnitude, f_{avg}, computed using Equation (7) and denoted by dashed lines, falls below the imposed lower limit, $f_{\text{avg}} < f_{\min}$, causing guidance to suggest a coasting segment (omitting the need for scaling to $f_{\max}$).



(a) NN guidance thrust magnitudes (b) Pre-Processing Steps ($\hat{x}$-$\hat{y}$,nondim)

Figure 10. Scenario 3: Neural network guidance step proceeds after initially computing a segment such that $f_1 < f_{\min}$. Total thrust time begins at 41.75 hours, reduces to 21.9 hours when combined, and results in 11.50 hours after adjustment to $f_{\max}$.

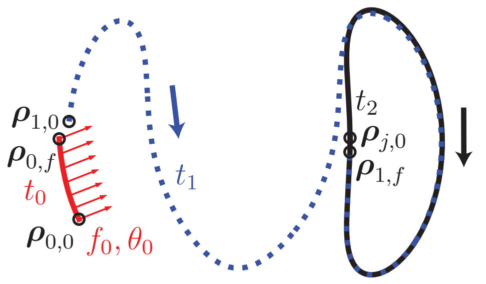

$\boldsymbol{\rho}_{1,0}$
$\boldsymbol{\rho}_{0,f}$
t_0
$\boldsymbol{\rho}_{0,0}$
f_0, θ_0
t_1
t_2
$\boldsymbol{\rho}_{j,0}$
$\boldsymbol{\rho}_{1,f}$

Figure 11. Targeting variables

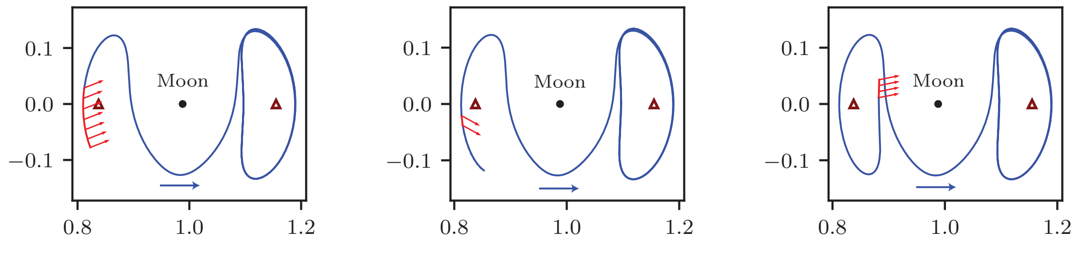
reference trajectory, $\boldsymbol{\rho}_{1,f}$, and the orange arrival periodic orbit in Figure 2(a). Assuming this orbit possesses a period t_2, and the nearest-neighbor state is $\boldsymbol{\rho}_2$, $\mathtt{n}$ revolutions are included in the initial guess, where $\mathtt{n} = 4$ is employed in this investigation. Stacking revolutions biases the converged path to maintain Lyapunov-like motion, however, does not ensure that the specific energy level is preserved. An alternative approach is to constrain the final state to be on the stable manifold, as detailed by Haapala and Howell.[5] While the stable manifold constraint approach preserves energy and periodicity, it poses challenges for a closed-loop system due to sensitivities to initial time parameters. Stacking revolutions is a more flexible approach if consistent closed-loop convergence is required.

To ensure that the thrust magnitude remains bounded, i.e., $f_0 \in [0, f_{\text{max}}]$, a parameterized thrust magnitude $\tilde{f}_0$ is introduced such that, $f_0 = \frac{1}{2} f_{\text{max}} \left(\sin \tilde{f}_0 + 1 \right)$. The new unbounded design variable $\tilde{f}_0$ is more easily included in the targeting process and guarantees $0 \leq f_0 \leq f_{\text{max}}$. An alternative approach introduces slack variables to enforce inequality constraints, however this method may diverge when f_0 is close to either bound. One benefit, or drawback, of employing $\tilde{f}_0$ is that no update is produced when $f_0 = 0$ or $f_0 = f_{\text{max}}$. The experiments in this investigation offer a guess $f_0 = f_{\text{max}}$ and, hence, thrust magnitude is not altered by targeting. Formulated as design variables, $\boldsymbol{X}$, and constraints, $\boldsymbol{F}(\boldsymbol{X})$, the targeting approach is summarized as,

$$\boldsymbol{X} = \Big(\underbrace{\tilde{f}_0 \quad \theta_0 \quad \boldsymbol{\rho}_0 \quad t_0}_{\text{Thrust Arc}} \quad \underbrace{\boldsymbol{\rho}_1 \quad t_1}_{\text{Ref. Traj.}} \quad \underbrace{\{\boldsymbol{\rho}_j \quad t_2\}_{j=2}^{\mathtt{n}+1}}_{\text{Arrival Orbit}} \Big)^T \qquad \boldsymbol{F}(\boldsymbol{X}) = \begin{pmatrix} \boldsymbol{\rho}_{0,f} - \boldsymbol{\rho}_{1,0} \\ \boldsymbol{\rho}_{1,f} - \boldsymbol{\rho}_{j,0} \\ \boldsymbol{\rho}_{j,f} - \boldsymbol{\rho}_{j+1,0} \\ \vdots \\ \boldsymbol{\rho}_{\mathtt{n},f} - \boldsymbol{\rho}_{\mathtt{n}+1,0} \end{pmatrix} \tag{18}$$

where Figure 11 depicts the "Thrust Arc" (red), the "Ref. Traj." (blue) and the "Arrival Orbit" (black). This targeter serves as a stand-in for any iterative approach that leverages a startup solution. The targeting scheme here is not intended to satisfy the strict requirements of flight software, but rather to illustrate an RL-generated controller and its incorporation into the initial guess generation process for existing corrections and optimization schemes. A neural network-enabled targeting guidance framework is an appealing hybrid approach, offering the neural network efficiency alongside targeting robustness to ensuring all mission criteria are satisfied.

The converged results for each representative scenario is plotted in Figure 12. Scenario 1 and 3 remain close to the initial guess, depicted in Figures 12(a) and 12(c). In both cases, the change in thrust angle is less than 0.1°. Thrust time increases by 5.6 hours for Scenario 1, and by 2.7 minutes for Scenario 3. The utility of thrust time scaling, detailed in Equation (16), is exemplified by the very small change in thrust time after Scenario 3 targeting. In contrast to Scenarios 1 and 3, the trajectory pre-processing step for Scenario 2 results in a 6 hour coasting segment due to the small combined thrust magnitude, as depicted in Figure 9(b). Each time a coast arc is suggested, the hybrid guidance approach is run from the beginning with the potential to compute a future thrust or coast segment. For Scenario 2, eight coasting segments are sequentially introduced, resulting in 2 days of coasting followed by a 7-hour thrust arc, depicted in Figure 12(b). Thrust magnitude is not updated for any scenario, since $f_0 = f_{\max}$. Together, the three representative scenarios illustrate how a hybrid guidance approach improves neural network-generated solutions, and avoids control choices that violate hardware constraints.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 12. Converged solutions for the three representative cases (x-y, nondim).

EXPERIMENTS

To evaluate the proposed hybrid guidance framework, 2000 randomly perturbed initial conditions are considered, where $3\sigma_r = 1000$ km and $3\sigma_v = 10$ m/s model the initial deviations. Each initial condition is simulated using the proposed hybrid guidance approach to achieve solutions similar to those in Figure 12. In analyzing the algorithm performance, the first, and most important, factor to consider is convergence, visualized in Figure 13. While the algorithm seeks to improve a neural network-generated path, cases arise where the startup solution is not sufficiently close to the converged solution, and the targeter diverges. For the simulated error level, the algorithm converges in the vast majority (98.15%) of simulations, and divergences for the other 1.85%. Next, the number of iterations is considered since rapid convergence is an important factor for potential onboard implementations. The targeter reaches a specified convergence threshold of 10^{-12} in 5 iterations or fewer in 42.3% of cases, and in 6 or fewer in 81.1% of trials, as depicted in Figure 13. The largest number of iterations reached across the 2000 trials is 15.

Each simulation is additionally evaluated using the NN guidance approach proposed in prior work,[4] e.g., the control profile depicted in Figure 3. Of the 2000 simulated condition, the neural network-driven solution reaches the vicinity of the target orbit in 99.4% of case (1988/2000). The hybrid targeting approach converges in only 2 of the 12 cases where the neural neural-only approach fails, indicating that the proposed hybrid approach is not able to consistently overcome situations where the neural network solution causes the spacecraft to diverge from the reference. Of the failure cases depicted in Figure 13, about 1 in 3 divergent cases correspond to simulations where the neural network-driven solution also fails.

Combined Solution Accuracy

Each simulation proceeds through the proposed pre-processing step and, hence, the accuracy of the computed thrust time and direction is considered. Accuracy is defined as the distance to the basin of convergence. Therefore, the converged solution differing significantly from the initial guess indicates that, for the set of conditions, the thrust segment combination step may be providing a poor estimate of the resulting solution. Conversely, a small change in value indicates a suitable startup solution.

Thrust direction does not significantly change in the majority of the 2000 simulations, indicating that the relationship for combined thrust direction, Equation (6), provides an accurate estimation for the given scenarios. In 94.5% of cases, the total change in thrust direction after targeting is less than two degrees, and less than four degrees in 98% of simulations. Thrust magnitude remains unchanged at $f_{\max}$ for all targeting experiments.

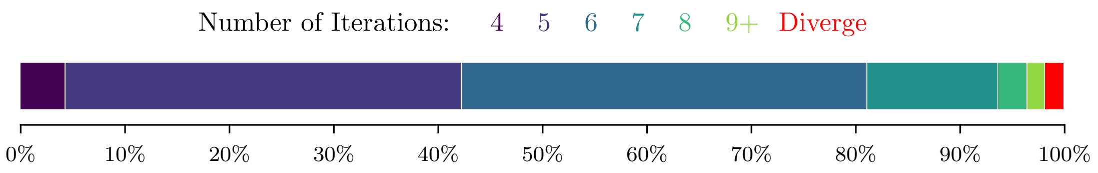


Figure 13. Number of iterations to convergence across 2000 simulations

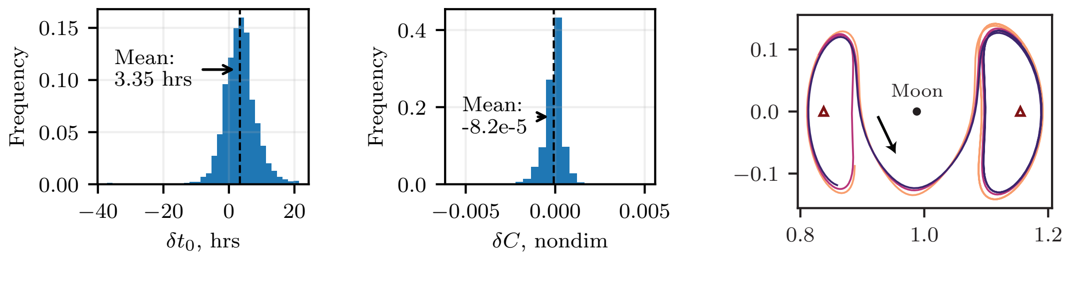
Total thrusting time is the most challenging quantity to approximate because it is not directly estimated by the neural network. The initial guess is computed as a weighted sum over the initial thrust segments, scaled for $f_0 = f_{\text{max}}$, as summarized in Equation (16). An empirical distribution for the overall change in thrust time, δt_0, over the 2000 simulations is plotted in Figure 14(a), where the average update is 3.35 hours. Figure 14(a) demonstrates that targeting causes thrust time to undergo a larger transformation than thrust direction and, hence, illustrates the degree of inaccuracy in the time estimate. In this case, the positive mean update value signifies an underestimation bias in the computed approximation. This bias may be addressed via alternative estimation methods, or employing a neural network architecture that directly evaluates time.

Energy Variation

The targeting scheme involves stacking revolutions of the arrival orbit and enforcing position and velocity continuity throughout the trajectory. While stacking revolutions biases the converged solution to resemble the desired Lyapunov orbit, it does not ensure that the target energy level is achieved. To determine the suitability of this approach, the resulting error in Jacobi constant for the 2000 trials is depicted in Figure 14(b), where δC is the difference in the Jacobi constant value between the reference orbit and the converged coasting segment after targeting. This empirical distribution demonstrates that the targeting process does not significantly alter energy, where the average case of $\delta C = -8.16 \times 10^{-5} \approx 0$ indicates no significant introduced energy bias. Furthermore, the simulations that undergo the largest change in energy still closely resemble the reference motion, depicted in Figure 14(c). While there is a small discernible difference in the amplitude of the resulting Lyapunov-like motion, this variation may be acceptable for a given application.

Mass Consumption

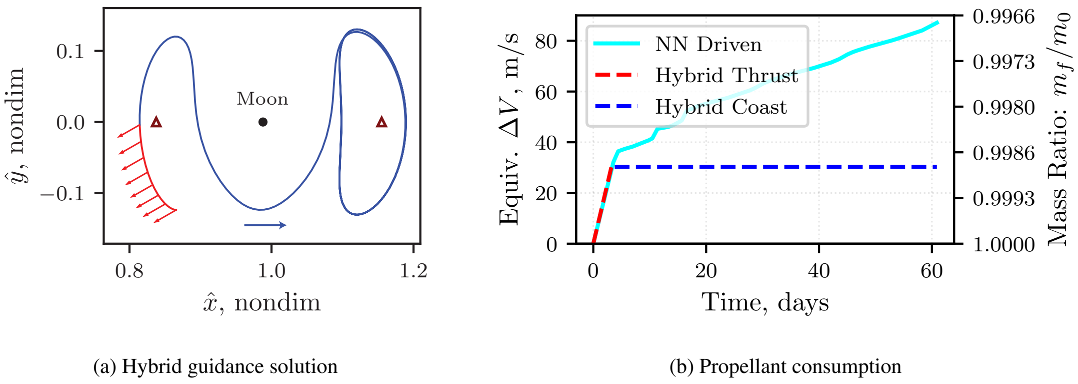
Solutions resulting from the proposed hybrid guidance strategy consume significantly less propellant than the previous[4] neural network guidance approach. The simulated neural network is not able to determine coasting segments and, hence, produces a large number of arcs at very low throttle levels. While propellant consumption across any one of these arcs remains small, the collected impact results in extremely propellant-



(a) Time updates (b) Energy errors (c) Maximum C changes ($\hat{x}$-$\hat{y}$, nondim)

Figure 14. Changes from reference motion over trials.

inefficient solutions. To illustrate the difference in mass consumption, consider the NN-driven solution depicted in Figure 3. This scenario serves as an initial guess for the hybrid guidance process, producing the converged trajectory in Figure 15(a). While similar in configuration space, these solutions differ significantly in propellant consumption, plotted in Figure 15(b). The mass change in the hybrid guidance solution closely resembles NN-driven solution for the first several days. However, these two approaches differ dramatically as the low-throttle arcs cause the NN-driven solution to continue increasing, resulting in more than double the propellant consumption of the hybrid algorithm during the 61-day simulation.



(a) Hybrid guidance solution

(b) Propellant consumption

Figure 15. Converged solution of the proposed guidance approach for the NN-driven scenario depicted in Figure 3. The hybrid solution (red and blue) demands significantly less propellant than the NN-driven result (cyan, Figure 3) over 61 days.

CONCLUDING REMARKS

Merging neural network and targeting paradigms offers a powerful hybrid guidance approach that simultaneously benefits from the speed of neural networks and the robustness of traditional iterative approaches to ensure all mission criteria are met. Neural network behavior is often unpredictable, which introduces error and risk into a potential machine learning-based guidance concept. In contrast, traditional iterative methods produce exact solutions, but are computational expensive and require accurate startup solutions. The proposed hybrid approach mitigates risk incurred by the neural network by employing differential corrections to ensure convergence and, conversely, speeds up the iterative approach by offering a rapid means to produce an initial guess for the resulting solution. Furthermore, the trajectory pre-processing component of the proposed guidance architecture enables the rapid combination and scaling of successive thrust segments to facilitate accurate startup solution generation despite any error introduced by the neural network control function.

ACKNOWLEDGMENTS

The authors thank the Purdue University School of Aeronautics and Astronautics for facilities and support in conducting this research. Special thanks to Dave Folta for useful conversations and feedback, and to Daniel Miller for valuable insights into machine learning. The authors acknowledge the MIT Supercloud and Lincoln Laboratory Supercomputing Center for providing HPC resources that have contributed to the research results reported within this paper. This work was supported by a NASA Space Technology Research Fellowship, NASA Grant 80NSSC19K1175.

REFERENCES

[1] B. Gaudet, R. Linares, and R. Furfaro, “Deep reinforcement learning for six degree-of-freedom planetary landing,” *Advances in Space Research*, Vol. 65, No. 7, 2020, pp. 1723–1741.

[2] B. Gaudet, R. Linares, and R. Furfaro, “Terminal adaptive guidance via reinforcement meta-learning: Applications to autonomous asteroid close-proximity operations,” *Acta Astronautica*, Vol. 171, 2020, pp. 1–13.

[3] A. Rubinsztejn, R. Sood, and F. E. Laipert, "Neural network optimal control in astrodynamics: Application to the missed thrust problem," *Acta astronautica*, Vol. 176, 2020, pp. 192–203.

[4] N. B. LaFarge, D. Miller, K. C. Howell, and R. Linares, "Autonomous Closed-Loop Guidance using Reinforcement Learning in a Low-Thrust, Multi-Body Dynamical Environment," *Acta Astronautica*, Vol. 186, Sept. 2021, pp. 1–23, https://doi.org/10.1016/j.actaastro.2021.05.014.

[5] A. F. Haapala and K. C. Howell, "A Framework for Constructing Transfers Linking Periodic Libration Point Orbits in the Spatial Circular Restricted Three-Body Problem," *International Journal of Bifurcations and Chaos*, Vol. 26, No. 5, 2016.

[6] B. Marchand, S. Scarritt, T. Pavlak, and K. Howell, "A dynamical approach to precision entry in multibody regimes: Dispersion manifolds," *Acta Astronautica*, Vol. 89, 2013, pp. 107–120.

[7] M. A. Vavrina, J. A. Englander, S. M. Phillips, and K. M. Hughes, "Global, Multi-Objective Trajectory Optimization With Parametric Spreading," *AAS/AIAA Astrodynamics Specialist Conference*, Stevenson, Washington, American Astronautical Society, Aug. 2017, pp. 1–20.

[8] J. D. Yencharis, R. F. Wiley, R. S. Davis, Q. A. Holmes, and K. T. Zeiler, "Apollo experience report: Development of guidance targeting techniques for the command module and launch vehicle," techreport, National Aeronautics and Space Administration, June 1972.

[9] B. G. Marchand, M. W. Weeks, C. W. Smith, and S. Scarritt, "Onboard Autonomous Targeting for the Trans-Earth Phase of Orion," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 3, 2010, pp. 943–956, 10.2514/1.42384.

[10] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning Data Mining, Inference, and Prediction*. Springer Series in Statistics, Springer, 2nd ed. ed., 2009.

[11] B. Dachwald, "Evolutionary Neurocontrol: A Smart Method for Global Optimization of Low-Thrust Trajectories," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Providence, Rhode Island, Aug. 2004, pp. 1–16.

[12] S. De Smet and D. J. Scheeres, "Identifying heteroclinic connections using artificial neural networks," *Acta Astronautica*, Vol. 161, Aug. 2019, pp. 192–199.

[13] N. L. O. Parrish, *Low Thrust Trajectory Optimization inCislunar and Translunar Space*. PhD dissertation, University of Colorado Boulder, 2018.

[14] R. Furfaro, A. Scorsoglio, R. Linares, and M. Massari, "Adaptive generalized ZEM-ZEV feedback guidance for planetary landing via a deep reinforcement learning approach," *Acta Astronautica*, Vol. 171, 2020, pp. 156–171.

[15] B. Gaudet, R. Linares, and R. Furfaro, "Six Degree-of-Freedom Hovering over an Asteroid with Unknown Environmental Dynamics via Reinforcement Learning," *20th AIAA Scitech Forum*, Orlando, Florida, AIAA, Jan. 2020, 10.2514/6.2020-1910.

[16] J. Broida and R. Linares, "Spacecraft Rendezvous Guidance in Cluttered Environments via Reinforcement Learning," *29th AAS/AIAA Space Flight Mechanics Meeting*, Ka'anapali, Hawaii, American Astronautical Society, Jan. 2019, pp. 1–15.

[17] B. Gaudet, R. Furfaro, and R. Linares, "Reinforcement learning for angle-only intercept guidance of maneuvering targets," *Aerospace Science and Technology*, Vol. 99, 2020.

[18] D. Guzzetti, "Reinforcement Learning And Topology Of Orbit Manifolds For Station-keeping Of Unstable Symmetric Periodic Orbits," *AAS/AIAA Astrodynamics Specialist Conference*, Portland, Maine, American Astronautical Society, Aug. 2019, pp. 1–20.

[19] J. A. Reiter and D. B. Spencer, "Augmenting Spacecraft Maneuver Strategy Optimization for Detection Avoidance With Competitive Coevolution," *20th AIAA Scitech Forum*, Orlando, Florida, AIAA, Jan. 2020, 10.2514/6.2020-1910.

[20] A. Das-Stuart, K. C. Howell, and D. C. Folta, "Rapid Trajectory Design in Complex Environments Enabled by Reinforcement Learning and Graph Search Strategies," *Acta Astronautica*, Vol. 171, 2020, pp. 172–195.

[21] D. Miller and R. Linares, "Low-Thrust Optimal Control via Reinforcement Learning," *29th AAS/AIAA Space Flight Mechanics Meeting*, Ka'anapali, Hawaii, American Astronautical Society, Jan. 2019, pp. 1–18.

[22] C. J. Sullivan and N. Bosanac, "Using Reinforcement Learning to Design a Low-Thrust Approach into a Periodic Orbit in a Multi-Body System," *20th AIAA Scitech Forum*, Orlando, Florida, AIAA, Jan. 2020, 10.2514/6.2020-1910.

[23] M. D. Rayman, P. Varghese, D. H. Lehman, and L. L. Livesay, "Results from the Deep Space 1 technology validation mission," *Acta Astronautica*, Vol. 47, No. 2, 2000, pp. 475–487.

[24] C. Russell and C. Raymond, *The Dawn Mission to Minor Planets 4 Vesta and 1 Ceres*. Springer, 1st ed. ed., 2012.

[25] W. Hart, G. M. Brown, S. M. Collins, M. De Soria-Santacruz Pich, P. Fieseler, D. Goebel, D. Marsh, D. Y. Oh, S. Snyder, N. Warner, G. Whiffen, L. T. Elkins-Tanton, J. F. Bell III, D. J. Lawrence, P. Lord, and Z. Pirkl, "Overview of the spacecraft design for the Psyche mission concept," *2018 IEEE Aerospace Conference*, Big Sky, Montana, IEEE, 2018, pp. 1–20.

[26] D. Irimies, D. Manzella, and T. Ferlin, "Summary of Gateway Power and Propulsion Element (PPE) Studies," *2019 IEEE Aerospace Conference*, Big Sky, Montana, IEEE, 2019, pp. 1–6.

[27] A. Cox, K. Howell, and D. Folta, "Dynamical structures in a low-thrust, multi-body model with applications to trajectory design," *Celestial Mechanics and Dynamical Astronomy*, Vol. 131, No. 3, 2019, pp. 1–34.

[28] A. D. Cox, *A Dynamical Systems Perspective For Preliminary Low-Thrust Trajectory Design In Multi-Body Regimes*. PhD dissertation, Purdue University, 2020.

[29] Busek Co. Inc., *BIT-3 RF Ion Thruster*, 2019.

[30] K. Nishiyama, S. Hosoda, K. Ueno, R. Tsukizaki, and H. Kuninaka, "Development and Testing of the Hayabusa2 Ion Engine System," *Transactions of the Japan Society for Aeronautical and Space Sciences*, Vol. 14, July 2016, pp. 131–140.

[31] D. Guzzetti, E. M. Zimovan, K. C. Howell, and D. C. Davis, "Stationkeeping Analysis for Spacecraft in Lunar Near Rectilinear Halo Orbits," *27th AAS/AIAA Space Flight Mechanics Meeting*, San Antonio, Texas, American Astronautical Society, Feb. 2017, pp. 1–24.

[32] J. M. Longuski, J. E. Prussing, and J. J. Guzmán, *Optimal Control with Aerospace Applications*, Vol. 32 of *Space Technology Library*. New York, NY: Springer New York, 2013.

[33] C. Ocampo, "Finite Burn Maneuver Modeling for a Generalized Spacecraft Trajectory Design and Optimization System," *Annals of the New York Academy of Sciences*, Vol. 1017, No. 1, 2004, pp. 210–233.

[34] H. B. Keller, *Numerical solution of two point boundary value problems*. CBMS-NSF regional conference series in applied mathematics, Philadelphia, Pa.: Society for Industrial and Applied Mathematics, 1976.