AAS 21-636

MULTI-OBJECTIVE REINFORCEMENT LEARNING FOR LOW-THRUST TRANSFER DESIGN BETWEEN LIBRATION POINT ORBITS

Christopher J. Sullivan[*], Natasha Bosanac[†], Alinda K. Mashiku[‡], and Rodney L. Anderson[§]

Multi-Reward Proximal Policy Optimization (MRPPO) is a multi-objective reinforcement learning algorithm used to construct low-thrust transfers between periodic orbits in multi-body systems. Previous implementations of MRPPO have relied on a predefined reference transfer to successfully train each policy. In this paper, an algorithmic modification labeled the 'moving reference', is introduced to autonomously construct these reference trajectories during training. With this modification, MRPPO is used to recover various low-thrust transfers between two periodic orbits in the Earth-Moon circular restricted three-body problem to solve a multi-objective optimization problem. These results are then compared with the solutions recovered via a gradient descent optimization scheme to validate the performance of MRPPO with the moving reference modification.

INTRODUCTION

Infusing autonomy into the trajectory design process may reduce the workload of human analysts and operators during mission concept development, and potentially enable a wide variety of mission concepts for spacecraft operating within multi-body systems. In multi-body systems, many existing approaches to trajectory design rely heavily on input from an expert trajectory designer: from a dynamical systems based method for initial guess construction to classical optimization methods.[1–6] However, designing a transfer in a multi-body system may be a time consuming process with the geometry of any initial guesses or governing parameters that are selected by an analyst often impacting the local basin of convergence and success of these numerical methods.[1,2] These limitations are particularly significant in multi-objective optimization problems with competing objectives and constraints, and in scenarios with little insight into the complex and diverse solution space. Motivated by these challenges, members of the astrodynamics community have recently begun to explore the use of reinforcement learning (RL) in constructing a framework for autonomous trajectory design in multi-body systems.

Reinforcement learning has previously been employed to autonomously construct a variety of transfers in multi-body systems.[7–14] One commonly used state-of-the-art RL algorithm, Proximal

[*]Ph.D. Student, Colorado Center for Astrodynamics Research, Smead Department of Aerospace Engineering Sciences, University of Colorado Boulder, Boulder CO, 80303.

[†]Assistant Professor, Colorado Center for Astrodynamics Research, Smead Department of Aerospace Engineering Sciences, University of Colorado Boulder, Boulder CO, 80303.

[‡]Deputy Manager, Navigation and Mission Design Branch, NASA Goddard Space Flight Center, Greenbelt, MD 20771

[§]Technologist, Mission Design & Navigation, NASA Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109

Policy Optimization (PPO), trains a policy to recover locally optimal behavior in unknown dynamical environments for a single objective function.[15] Multi-Reward Proximal Policy Optimization (MRPPO), developed by Sullivan and Bosanac, uses PPO as a foundation within a multi-objective framework to simultaneously train multiple policies, each maximizing unique objective functions using information gained from the experiences of all agents.[9] Due to this architecture, MRPPO retains the beneficial convergence properties of PPO in chaotic environments, while also reducing the computational time required to recover distinct components of the solution space in a multi-objective optimization problem.[9,12–14,16] However, standard implementations of PPO and MRPPO have been observed in our previous work to struggle to converge on complex transfers between two periodic orbits without a predefined initial guess or reference trajectory; such limitations are consistent with the challenges of using RL for complex path planning in a broader array of disciplines.[9,10,17–20] This observation motivates the algorithmic modification presented in this paper.

An algorithmic modification, labeled the 'moving reference', is constructed to enable the autonomous generation of reference trajectories during training of MRPPO. As observed in our previous work, reference trajectories often serve as valuable guides for training policies to recover transfers between two periodic orbits; without a reference trajectory, training in both PPO and MRPPO may fail to produce successful policies.[9,10] As constructed, the moving reference modification requires no initial guess to be defined prior to training and enables the best trajectories generated throughout training for each policy to serve as the moving reference until a better trajectory is generated. Using MRPPO with the moving reference modification enables the design of transfers in high-dimensional, multi-objective solution spaces.

This paper focuses on exploring the moving reference modification within MRRPO to design low-thrust transfers for SmallSats in the Earth-Moon system; both extending and addressing some limitations of earlier implementations of MRPPO. This algorithmic modification is demonstrated in the context of transfers between distinct libration point orbits, specifically transiting from an L_1 northern halo orbit to an L_2 southern halo orbit, in the Earth-Moon circular restricted three-body problem (CR3BP). Once the periodic orbits, spacecraft parameters, and reward function are defined, MRPPO with the moving reference modification autonomously recovers solutions without any additional information required from a trajectory designer nor machine learning expert. Applying MRPPO with the moving reference modification to a multi-objective optimization problem and balancing competing objectives, such as flight time and propellant mass usage, results in the recovery of a subset of the multi-objective solution space. Additionally, the validity of the results generated by MRPPO with the moving reference modification is examined in this paper via a comparison to the nearby local optima recovered using gradient descent optimization. Then, the solution spaces developed by both methodologies are analyzed to validate the results produced by MRPPO with the moving reference modification.

DYNAMICAL MODEL

Often in preliminary trajectory design, approximate dynamical models are used to reduce the complexity of the design space while still retaining the fundamental dynamical properties of the system. For instance, the CR3BP may be used to approximate the motion of a spacecraft under the gravitational influences of two larger primary bodies such as the Earth and Moon. The equations of motion for the CR3BP may be modified to include additional accelerations imparted by a spacecraft's propulsion system. In this paper, the low-thrust CR3BP equations of motion are used to model the motion of a low-thrust-enabled spacecraft in cislunar space.

Circular Restricted Three-Body Problem

The CR3BP is used to model the natural motion of a spacecraft in the Earth-Moon system subject to the gravitational influences of the two primary bodies. This autonomous dynamical model approximates the motion of the primaries, with masses M_1 and M_2 respectively, modeled as point masses following circular orbits about their mutual barycenter. In this model, a rotating frame, $(\hat{x}, \hat{y}, \hat{z})$, is defined using the primaries: the $\hat{x}$ axis is directed from the Earth to the Moon, the $\hat{z}$ axis is defined with the orbital angular momentum vectors of the primaries, and the $\hat{y}$ axis is defined to form a right-handed triad.[21] Parameters are typically nondimensionalized via the characteristic quantities l^*, m^*, and t^*.[21] Table 1 displays the characteristic quantities for the Earth-Moon CR3BP used in this paper. Nondimensionalizing the position, velocity, time, and mass quantities typically reduces the potential for poor conditioning in numerical integration, facilitates a smoother training process for neural networks, and enables comparisons between distinct multi-body systems. Then, the nondimensionalized state of the spacecraft is defined as $\bar{x} = [\bar{d}^T, \bar{v}^T]^T$ where $\bar{d} = [x, y, z]^T$ and $\bar{v} = [\dot{x}, \dot{y}, \dot{z}]^T$ are the position and velocity vectors in the rotating frame with respect to the barycenter of the system. Additionally, the mass ratio of the system, a quantity that significantly influences the solution space in the CR3BP, is defined as $\mu = M_2/(M_1 + M_2)$.[21] Using these definitions, the equations of motion that govern the motion of a spacecraft in the CR3BP are written as

$$\ddot{x} - 2\dot{y} = \frac{\partial U^*}{\partial x} \quad \ddot{y} + 2\dot{x} = \frac{\partial U^*}{\partial y} \quad \ddot{z} = \frac{\partial U^*}{\partial z} \tag{1}$$

where the pseudo-potential function is $U^* = (1/2)(x^2+y^2) + (1-\mu)/d_1 + \mu/d_2$ and the distances from the spacecraft to Earth and the Moon are, respectively, $d_1 = \sqrt{(x+\mu)^2 + y^2 + z^2}$ and $d_2 = \sqrt{(x-1+\mu)^2 + y^2 + z^2}$.[21] In this dynamical model, a constant of integration, denoted the Jacobi constant exists and is equal to $C_J = 2U^* - \dot{x}^2 - \dot{y}^2 - \dot{z}^2$.[21] The Jacobi constant is valuable for describing the relative energy between natural solutions to the CR3BP and also the states along a maneuver-assisted transfer.

Table 1: Earth-Moon CR3BP characteristic quantities.[22]

Parameter	Value
Characteristic length, l^*	384,400 km
Characteristic time, t^*	375,132 s
Characteristic mass, m^*	6.0477×10^{24} kg
Mass ratio, μ	1.2151×10^{-2}

Fundamental solutions to the CR3BP include equilibrium points, periodic orbits, and hyperbolic invariant manifolds. Five equilibrium points, denoted L_1-L_5, exist in the CR3BP with each possessing nearby periodic orbit families of varying energy levels and inclinations.[22] In the Earth-Moon system, L_1 and L_2 are of particular interest due to their proximity to the Moon and the periodic orbit families that emanate from the points.[23] Two families of periodic orbits, the L_1 northern halo orbit family and L_2 southern halo orbit family, each have members that offer extensive coverage of the lunar surface and supply constant line-of-sight with the Earth.[23] In this paper, a variety of low-thrust transfers are constructed from an L_1 northern halo orbit to an L_2 southern halo orbit with Jacobi constants of 3.15 and 3.11, respectively, within the multi-objective solution space associated with propellant mass usage and flight time.

Incorporating Low-Thrust Acceleration into the CR3BP

Spacecraft equipped with low-thrust propulsion systems may induce additional accelerations influencing their motion within the multi-body system. To account for the low-thrust acceleration, the CR3BP equations of motion are modified and a fourth differential equation is added to model the propellant mass usage. In this paper, the spacecraft is assumed to be equipped with a single variable thrust, constant specific impulse engine whereby the thrust direction is expressed in the Earth-Moon rotating frame. The nondimensional acceleration imparted by the low-thrust engine is written in the Earth-Moon rotating frame as

$$\bar{a} = \frac{Tt^{*2}}{1000 m_{s/c} m_0 l^*} \left(\tilde{u}_x \hat{x} + \tilde{u}_y \hat{y} + \tilde{u}_z \hat{z} \right) = a_x \hat{x} + a_y \hat{y} + a_z \hat{z} \tag{2}$$

where T is the thrust magnitude, m_0 is the initial wet mass of the spacecraft in dimensional units, $m_{s/c}$ is the mass of the spacecraft nondimensionalized by the initial wet mass, and $\bar{\tilde{u}} = [\tilde{u}_x, \tilde{u}_y, \tilde{u}_z]^T$ is the thrust direction vector. The mass flow rate equation is incorporated into the equations of motion to reflect the propellant mass usage of the spacecraft. With the low-thrust engine activated, the equations of motion for the low-thrust-enabled CR3BP are written as

$$\ddot{x} - 2\dot{y} = \frac{\partial U^*}{\partial x} + a_x \quad \ddot{y} + 2\dot{x} = \frac{\partial U^*}{\partial y} + a_y \quad \ddot{z} = \frac{\partial U^*}{\partial z} + a_z \quad \dot{m}_{s/c} = \frac{-T|\bar{\tilde{u}}|t^*}{I_{sp} g_0 m_0} \tag{3}$$

where I_{sp} is the specific impulse, $|\bar{\tilde{u}}|$ is the magnitude of the thrust vector, and $g_0 = 9.81$ m/s^2 is the gravitational acceleration at the surface of the Earth.[22] These four differential equations are used to model the dynamics of a low-thrust-enabled spacecraft in the Earth-Moon CR3BP.

DEEP REINFORCEMENT LEARNING

Recently, RL algorithms have been successfully applied in a variety of applications to solve complex, high-dimensional optimizations problems, including within the astrodynamics community.[7–14,24–30] A majority of these RL applications train neural networks, which act as universal function approximators, to learn optimal behavior via mapping actions, denoted as $\bar{a}_t$, to states, represented by $\bar{s}_t$, that maximize a set of objectives within an unknown environment.[31] Additionally, neural networks are able to learn the complex, high-dimensional state-action relationships within an environment much more robustly and efficiently than traditional RL techniques.[32] The neural networks are parameterized by a number of hidden layers and hidden nodes, each with weights, biases, and activation functions. In this investigation, the Rectified Linear Unit (ReLU) is selected as the activation function while the weights and biases of each connection within the neural networks are encapsulated within the parameter vector, $\bar{\theta}$, which is updated throughout training. Often, the neural networks are initialized without any knowledge of the environment or objectives; as a result, they may recover optimal behavior without biasing from a human designer. However, neural networks with two or more hidden layers are typically limited to recovering a local minima rather than the global minima for a set of objectives.[33,34] These objectives are encapsulated within the reward function which mathematically determines the instantaneous reward of any state-action pair within the environment.[35] Then, the goal of RL algorithms is to train the neural networks to maximize the expected total reward within the environment for a single trajectory.[36] To do so, state-of-the-art RL algorithms are constructed using neural networks that control one or more agents within the environment.[37] The agents explore the environment and collect state-action-reward experiences that are used to update the neural networks.[38,39] Using this formulation, RL algorithms may be applied to

a variety of applications including optimization problems where traditional optimization techniques cannot be applied to uncover optimal behavior or where significant human input is required.[31,40]

State-of-the-art RL algorithms often incorporate an actor-critic structure to train two neural networks to uncover locally optimal behavior within the environment. In an actor-critic structure, learning is separated into two distinct components: (a) the actor maps optimal actions to every state in the environment, denoted as the policy function $\pi(\bar{s}_t, \bar{u}_t)$, and (b) the critic estimates the value function, denoted as $V^{\pi}(\bar{s}_t)$, for every state in the environment.[41] The value function determines the expected discounted cumulative reward for every state in the environment and is computed as

$$V^{\pi}(\bar{s}_t) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t(\bar{s}_t, \bar{u}_t, \bar{s}_{t+1})\right] \tag{4}$$

where γ denotes the discount factor which reduces the influence of future rewards and $r_t(\bar{s}_t, \bar{u}_t, \bar{s}_{t+1})$ represents the reward function for the state and action at time t and the state at time $t+1$.[42] Using two distinct structures to learn the optimal behavior simplifies the learning process and leads to more robust convergence properties compared to singular structure methods such as policy iteration or value iteration methods.[41] Additionally, while many function approximators exist, neural networks have demonstrated advantageous performance across many applications; thus, they are used in this paper to construct the actor and the critic. While actor-critic structures with neural networks are often used as the basis for learning the optimal policy and value functions, there are a variety of algorithms available for training the networks.

PPO is one RL algorithm that has demonstrated strong performance and favorable convergence properties in chaotic environments.[16,43] PPO trains a single policy to maximize the total reward returned for an environment and reward function while limiting the size of updates from destabilizing the networks by enforcing a soft constraint within an objective function.[15] This objective function is composed of three elements: (1) a clipped objective that limits the influence of highly rewarding or penalizing state-action pairs from over correcting the networks, (2) a value error term that computes the error in the estimation of the value function, and (3) an entropy term that encourages exploration in the environment.[15] The clipped objective is defined as

$$L_t^{CLIP}(\bar{\theta}_j) = \hat{\mathbb{E}}_t[\min(R_t(\bar{\theta}_j)\hat{A}_t, \text{clip}(R_t(\bar{\theta}_j), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)] \tag{5}$$

where

$$R_t(\bar{\theta}_j) = \frac{\pi_{\bar{\theta},j}(\bar{u}_t|\bar{s}_t)}{\pi_{\bar{\theta},j-1}(\bar{u}_t|\bar{s}_t)} \tag{6}$$

is the standard definition of the probability ratio between the old and new policies such that ratios near unity correspond to small changes in the policy, ε denotes the clipping parameter controlling how far the new policy may deviate, and $\hat{A}_t^{\pi}$ is the estimated advantage function.[15] The advantage function uses Generalized Advantage Estimation (GAE) to evaluate how beneficial a state-action pair is based on the expected and returned rewards and is calculated as

$$\hat{A}_t^{\pi}(\bar{s}_t, \bar{u}_t) = \sum_{\ell=0}^{\infty} (\gamma\lambda)^{\ell} \partial_{t+\ell} \tag{7}$$

where

$$\partial_t = r_t(\bar{s}_t, \bar{u}_t, \bar{s}_{t+1}) + \gamma V^{\pi}(\bar{s}_{t+1}) - V^{\pi}(\bar{s}_t) \tag{8}$$

is the estimated advantage of action $\bar{u}_t$ and λ is defined as the GAE factor which influences the bias-variance trade-off within the estimated advantages.[36] The clipped objective is combined with the value error and entropy components to create the objective function used in PPO and written as

$$L_t^{CLIP+VF+S}(\bar{\theta}_j) = \hat{\mathbb{E}}_t[L_t^{CLIP}(\bar{\theta}_j) - c_1 L_t^{VF}(\bar{\theta}_j) + c_2 S[\pi_{\bar{\theta},j}](\bar{s}_t)] \tag{9}$$

where $L_t^{VF}(\bar{\theta}_j) = (V_{\bar{\theta}}^{est}(\bar{s}_t) - V_t^{act})^2$ is the squared value error term, $S[\pi_{\bar{\theta},j}](\bar{s}_t)$ denotes the entropy term, and c_1 and c_2 are scalar coefficients for the value error and entropy terms, respectively.[15] To train the networks to optimize this objective function, the AdamW optimizer is selected, which offers improved convergence performance compared to the traditionally used Adam optimizer by incorporating a weight decay term in the update to the neural networks.[44] Then, using PPO as a foundation, MRPPO constructs a multi-objective framework that trains multiple policies simultaneously while still retaining PPO's beneficial convergence properties in chaotic environments.[9]

Multi-Reward Proximal Policy Optimization

MRPPO is a multi-objective framework that leverages PPO to train multiple policies, each with a distinct reward function, simultaneously by sharing state-action propagation data generated by one policy with every other policy. Sharing state-action pairs across policies enables the same amount of environmental data to be used to train multiple policies. This structure significantly reduces the computational time and effort required to generate a variety of solutions in a multi-objective solution space. Specifically, MRPPO trains N policies denoted $[\pi_1 \dots \pi_N]$, each controlling K_i independent agents in the environment, to maximize unique objective prioritizations defined via the corresponding reward functions $[r_{1,t} \dots r_{N,t}]$. In each case, the policies aim to maximize the total reward generated for their assigned reward functions by learning the state transitions, system dynamics, and locally optimal behavior. Once the state-action propagation data is generated for the policies, each policy feeds the data through its unique reward function to produce state-action-reward experiences that are used to update the policy. Figure 1 depicts the architecture of MRPPO along with the flow of information through the system whereby N policies in blue each control K_i agents in red, state-action pairs are propagated within the dynamical model to generate the next state, the propagation information is stored in the shared memory in pink, and finally at the update phase to the networks, the entire set of state-action pairs from all policies are input to each reward function to generate the unique state-action-reward experiences. However, due to the limited machine precision available and the statistical nature of certain actions, the objective function used by PPO must be adjusted for MRPPO to enable a smooth training process.

State-action-reward experiences are used to update the networks encoding the policies by evaluating the objective function. However, since policies become more exploitative throughout training as the policies learn more information about the environment and objectives, certain actions undertaken by other policies may become highly statistically unlikely. This effect is compounded by limited floating point precision leading to the probability ratio defined in Eq. (6) approaching infinity. To negate this effect, the policy ratio for policy $\pi_{\bar{\theta},i,j}$ is redefined as

$$R_{i,t}(\bar{\theta}_{i,j}) = \pi_{\bar{\theta},i,j}(\bar{u}_t|\bar{s}_t) - \pi_{\bar{\theta},i,j-1}(\bar{u}_t|\bar{s}_t) \tag{10}$$

whereby ratios near zero now correspond to small changes between the old and new policies.[45,46] Then, the clipping objective from Eq. (5) must be redefined to enforce ratios near zero via

$$L_{i,t}^{CLIP}(\bar{\theta}_{i,j}) = \hat{\mathbb{E}}_t[\min(R_{i,t}(\bar{\theta}_{i,j})\hat{A}_{i,t}, \text{clip}(R_{i,t}(\bar{\theta}_{i,j}), -\varepsilon, \varepsilon)\hat{A}_{i,t})] \tag{11}$$

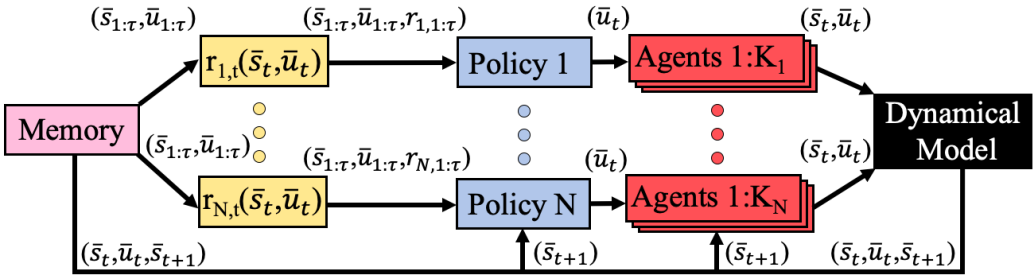


Figure 1: Conceptual representation of MRPPO illustrating N policies with assigned reward functions $r_{i,t}(\bar{s}_t, \bar{u}_t)$ commanding K_i agents in a common dynamical model.

MRPPO is implemented in PyTorch, an open-source machine learning library.[47] Additionally, MRPPO retains PPO's favorable convergence properties and robustness to hyperparameter selection.[9,16,43] Table 2 displays the hyperparameters and construction parameters used in this paper. Specifically, a grid search exploration by Sullivan and Bosanac in combination with tuning in a distinct trajectory design scenario is used to select these hyperparameters.[9,16,43] Additionally, both MRPPO and PPO are both robust to hyperparameter selection compared to other state-of-the-art RL algorithms.[9,16,43] Using this set of hyperparameters, probability ratio, and clipping objective for MRPPO enables multiple policies to be trained simultaneously without the threat of destabilization due to limited machine precision.

Table 2: MRPPO hyperparameter and construction parameter values.

Quantity	Value	Quantity	Value
Update Phases	500	Clipping Parameter, ε	2×10^{-2}
Environmental Steps, τ	4096	Initial Learning Rate, l_r	1×10^{-3}
Epochs, E	5	Actor Hidden Layers	2
Mini Batches, M	4	Actor Nodes per Hidden Layer	64
Discount Factor, γ	0.95	Critic Hidden Layers	1
GAE Factor, λ	0.9	Critic Nodes per Hidden Layer	1024
Value Function Coefficient, c_1	0.5	Activation Function	ReLU
Entropy Coefficient, c_2	1×10^{-3}	Agents per Policy	4

Moving Reference Modification

An algorithmic modification, labeled the 'moving reference', is constructed to enable the autonomous generation of reference trajectories during training of MRPPO. As observed in previous work, reference trajectories often serve as valuable guides for training policies to recover transfers between two periodic orbits; without a reference trajectory, training in both PPO and MRPPO may fail to produce successful policies.[9,10] The moving reference modification is defined by the following steps during the training of each policy:

1. Generate a trajectory for policy P_i
2. Evaluate as reference trajectory for policy P_i
    - If first trajectory: set as the current reference trajectory for policy P_i
    - Else: compare to existing reference trajectory for policy P_i and replace if better

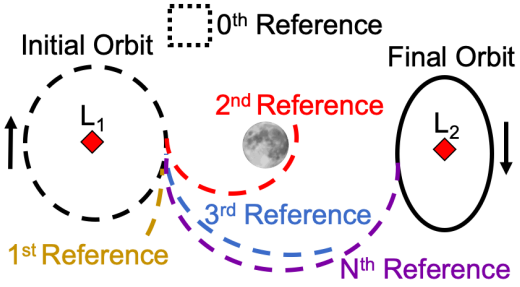


Figure 2: Updating the reference trajectory throughout training in a transfer design scenario.

Determining whether a new trajectory supplies a better reference may depend on the properties of interest, e.g., via criteria defined in the phase space, evaluation of the reward function, etc. In this paper, reference trajectories are initially selected based on their closest approach to the final periodic orbit over several time steps and, at later updates, based on a combination of the flight time and propellant mass usage. Specifically, once the average distance for the 10 closest states falls below 10,000 km, reference trajectories are then selected based on maximizing $L = \sum_{t=1}^{t_F} -4 - c_m \Delta m_t$ where c_m is the propellant mass usage coefficient assigned to each policy. Of course, using a single reference trajectory for each policy influences the geometry of solutions from other departure locations along the periodic orbit; addressing this limitation is the focus of ongoing work. Figure 2 illustrates the moving reference concept during training to recover transfers between two periodic orbits. Each policy produces a unique reference trajectory corresponding to the distinct objective prioritization assigned to the policy through the reward function. As constructed, the moving reference modification requires no initial guess to be defined prior to training and enables the best trajectories generated throughout training for each policy to serve as the reference until a better trajectory is generated.

RECOVERING LOW-THRUST TRANSFERS USING MRPPO

MRPPO with the moving reference modification is used to recover low-thrust transfers in the Earth-Moon CR3BP without the need for a predefined reference trajectory or initial guess. In this paper, four policies are initialized without any a priori knowledge of the environment nor objectives and are trained to guide a low-thrust-enabled SmallSat from an initial periodic orbit to a final periodic orbit. Then, these four policies are evaluated to uncover a subset of the propellant mass usage and flight time trade space.

Trajectory Design Scenario

A sample trajectory design scenario is defined for a low-thrust transfer between two periodic orbits in the Earth-Moon CR3BP. In this scenario, the initial periodic orbit is a low-energy L_1 northern halo orbit whereas the final orbit is a higher-energy L_2 southern halo orbit; a scenario previously explored in our past work due to the significant interest surrounding these transfer types.[10,23] Table 3 presents initial conditions and characteristics of the periodic orbits. Due to the difference in energy between the orbits, maneuvers are required to complete a transfer in the CR3BP. Furthermore, the low-thrust transfers are expected to resemble the invariant manifold structures that emanate from both orbits and admit a gradual increase in energy along the transfer.[10,48]

Table 3: Nondimensional characteristics of the selected initial and final periodic orbits.

Orbit	x	y	z	$\dot{x}$	$\dot{y}$	$\dot{z}$	**Period**	C_J
L_1 Northern Halo	0.8687	0	-0.0451	0	-0.1881	0	2.7614	3.15
L_2 Southern Halo	1.1676	0	-0.1029	0	-0.1973	0	3.3216	3.11

Spacecraft Properties and Initial Conditions

The agents in the environment are modeled as ESPA-class SmallSats with an initial wet mass of 180 kg equipped with a single variable thrust, constant specific impulse engine with full control over thrust direction in the Earth-Moon CR3BP.[49] The maximum available thrust of the engine is $T_{max} = 0.15$ N while the specific impulse is assumed as $I_{sp} = 3000$ s, both within the capabilities of current or near-future low-thrust engines.[49,50] During training, the initial conditions for the spacecraft are randomly drawn from along the initial periodic orbit and perturbed in both position and velocity components. The induced perturbations are drawn from a Gaussian distribution with zero mean and a standard deviation of 10^{-3} nondimensional units for both the position and velocity. However, when evaluating the trained neural networks, a common set of 1,000 perturbed initial conditions is used to facilitate more equivalent comparisons between the policies. This perturbed initial condition formulation reflects off-nominal conditions and enables the networks to experience a wider array of environmental information surrounding the initial periodic orbit. Once the spacecraft are initialized and initial conditions are drawn, the networks generate actions and gather state-action-reward experiences used to update the policies.

State and Action Definitions

Once an initial condition is selected for an agent, the spacecraft information is used to form a state vector that is input to the neural networks to generate an action. The state vector for the neural networks is defined in this paper as a 15×1 vector written as

$$\bar{s}_t = [\bar{d}_t^T, \bar{v}_t^T, m_{s/c}, \Delta\bar{d}_t^T, \Delta\bar{v}_t^T, \kappa, t]^T \tag{12}$$

In this state formulation, $\bar{d}_t$ and $\bar{v}_t$ denote the absolute state of the spacecraft in the Earth-Moon CR3BP; $m_{s/c}$ is the nondimensionalized spacecraft mass; $\Delta\bar{d}_t$ and $\Delta\bar{v}_t$ represent the relative states of the spacecraft measured with respect to the initial orbit, final orbit, or reference trajectory; κ signifies whether the relative states are measured with respect to the initial orbit, final orbit, or reference trajectory, and t is the current time step. Specifically, the relative states are measured with respect to the closest dynamical structure or reference trajectory in position space unless the spacecraft is within 10,000 km of the final orbit. Then, the final orbit is used to prevent chattering in the relative states between a reference trajectory and final orbit. Then, κ corresponds to how the relative states are measured and is set as -1, 0, or 1 for the initial orbit, reference trajectory, or final orbit, respectively. This state formulation provides the networks information about the spacecraft and enables the trained networks to draw locally optimal actions for the spacecraft.

Once the current state is input to the neural networks, the actor neural network generates an action that aims to maximize the long-term reward in the environment. The action vector is a 4×1 vector written as $\bar{u}_t = [u_x, u_y, u_z, u_T]$ where the first three components determine the thrust direction and the fourth component determines the thrust magnitude. The first three components are normalized to produce the thrust direction vector, $\bar{\hat{u}}$, used in Eq. (2). Then, to produce an actionable thrust

magnitude, T, for Eq. (2), the thrust magnitude component, u_T, is scaled using

$$T = T_{max}\left(\frac{\tanh(u_T)+1}{2}\right) \tag{13}$$

using the hyperbolic tangent function first to constrain the thrust component to be between -1 and 1 and then normalized to ensure that the thrust magnitude is between 0 and T_{max}. Once a state-action pair is generated, the spacecraft is propagated forward in time using the low-thrust-enabled CR3BP equations of motion for a time step of $\Delta t = 6 \times 10^{-2}$ nondimensional time units, or approximately 6 hours. State-action pairs are continually generated along a single trajectory until a termination condition is activated and the spacecraft is reset.

Reward Function Formulation

Using MRPPO with the moving reference modification, each policy possesses a distinct reward function that significantly influences the resulting behavior of the policy. In this paper, the reward functions are structured to incentivize the policies to guide the spacecraft from the initial orbit to the final orbit by approximately following the generated reference trajectories while also reducing propellant mass usage. The reward functions are structured based on whether the spacecraft is closest to the initial orbit, final orbit, or reference trajectory. However, if the spacecraft is within 10,000 km of the final orbit, then the specification defaults to select the final orbit configuration. The general reward function formulation for this trajectory design scenario is written as

$$r_t(\bar{s}_t, \bar{u}_t) = \begin{cases} -8 - 10|\Delta\bar{d}_{t+\Delta t}| - |\Delta\bar{v}_{t+\Delta t}| - c_m(m_{s/c,t} - m_{s/c,t+\Delta t}) + \Omega & \text{Initial Orbit} \\ -4 - 10|\Delta\bar{d}_{t+\Delta t}| - |\Delta\bar{v}_{t+\Delta t}| - c_m(m_{s/c,t} - m_{s/c,t+\Delta t}) + \Omega & \text{Reference} \\ -100|\Delta\bar{d}_{t+\Delta t}| - 10|\Delta\bar{v}_{t+\Delta t}| - c_m(m_{s/c,t} - m_{s/c,t+\Delta t}) + \Omega & \text{Final Orbit} \end{cases} \tag{14}$$

where c_m denotes the mass coefficient which scales the penalty on propellant mass usage and Ω is the bonus or penalty associated with the termination conditions. The coefficients on the state discontinuity terms are selected to encourage the policies to stay close to the initial orbit and reference trajectory. However, when the policies approach the final orbit, the coefficients are increased to incentivize convergence to the final orbit. This formulation enables the policies to learn to follow the dynamical structures within the system while also encouraging the targeting the final orbit.

Scaling the penalty on propellant mass usage enables the recovery of a variety of solutions within the flight time and propellant mass usage trade space. The four policies trained on this trajectory design scenario are assigned mass coefficients of $c_m = [0, 83.33, 166.66, 250]$ whereby a higher mass coefficient corresponds to a higher penalty on propellant mass usage. However, unlike in a conventional optimization problem, the reward function cannot be set to maximize the final mass or to have too high of a mass coefficient because the policies would then be incentivized to not perform any actions and stay along the initial orbit.

Once a termination condition has activated corresponding to the end of a trajectory, a bonus or penalty is applied depending on whether the trajectory converged upon the final orbit. In this scenario, four termination conditions are defined: (1) the spacecraft is within 5×10^{-3} in position and velocity of the closest state in position space of the final halo orbit and $\Omega = 1000$, (2) the maximum number of time steps, 150, is reached and $\Omega = 0$, (3) the spacecraft strays over 12,500 km from the initial orbit, final orbit, and reference trajectory and $\Omega = -1000$, and (4) the spacecraft enters within 5 Moon radii and $\Omega = -1000$. These penalties discourage the spacecraft from departing the system or impacting the Moon while the bonus rewards convergence to the final orbit.

NONLINEAR CONSTRAINED OPTIMIZATION

Once locally optimal solutions are recovered using MRPPO, the solutions are compared to those generated by a classical optimization technique. This comparison is used to assess the results of MRPPO and compare the solution spaces uncovered by the two methodologies. Since most optimization techniques require an initial guess, the solutions developed by MRPPO serve as the initial guess prior to optimization. In this paper, a multiple shooting algorithm in combination with *fmincon* in MATLAB© is used to explore whether the solutions generated by MRPPO lie close to locally optimal solutions.[51] The characteristics of the initial guess, including the low-thrust propagation time, maximum available thrust, and thrust vector formulation, are consistent between the MRPPO and *fmincon* implementations, but the limitations of *fmincon* prevent a completely equivalent implementation.[51] Namely, *fmincon* requires the objective function and constraints to be continuous and possess continuous first order derivatives. Then, the objective function for *fmincon* must be reformulated to solely maximize the final mass while state and action continuity is enforced via constraints. While the recovered solution spaces are expected to be slightly different, the comparison facilitates insight into the validity of the solution space recovered by MRPPO with the moving reference modification.

Free Variables

A multiple shooting algorithm is used to constrain the solutions that are optimized using *fmincon*. The free variable vector describes two distinct components of the trajectory: (1) a low-thrust segment based on a reference trajectory produced by MRPPO and (2) a coast segment along the final periodic orbit. These segments are discretized into multiple arcs that are described by nodes located at their initial states. The initial node is defined at the perturbed initial condition of the reference trajectory. However, only the action components for this node, $\bar{u}_1$, are included as free variables in the free variable vector. Then, all proceeding low-thrust nodes are written as $\bar{N}_{LT,i\geq 2} = \left[x, y, z, \dot{x}, \dot{y}, \dot{z}, m_{s/c}, u_x, u_y, u_z, T\right]^T$. For the final orbit coast segment, the position, velocity, and propagation time are all included as free variables and written as $\bar{N}_C = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \Delta t]^T$. Unlike the coast nodes, the free variables for the low-thrust nodes do not include the propagation time to facilitate a more direct comparison to the MRPPO formulation and ensure a consistent flight time. Finally, the entire free variable vector is written as $\bar{X} = \left[\bar{u}_1, \bar{N}_{LT,2}, \ldots, \bar{N}_{LT,n}, \bar{N}_{C,1}, \ldots \bar{N}_{C,q}\right]^T$. where n denotes the number of low-thrust arcs and q is the number of coast arcs along the final periodic orbit. This free variable vector formulation is equivalent to the MRPPO with moving reference modification implementation.

Equality Constraints

Unlike the MRPPO implementation where soft constraints are typically incorporated via penalties in the reward function, hard constraints are explicitly defined and satisfied by the optimal solutions. The constraint vector for this trajectory design scenario is formulated to ensure full state continuity between all nodes, mass continuity between low-thrust nodes, and unity magnitude for the thrust direction for each low-thrust node. Full state continuity is enforced via $\bar{c}_i = \bar{x}_{i,t+\Delta t} - \bar{x}_{i+1,t}$ where i includes all nodes except for the final node. The full state continuity constraint for the final node to ensure insertion into the final orbit is written as $\bar{c}_F = \bar{x}_{p+n,t+\Delta t} - \bar{x}_{IC,F}$ where $\bar{x}_{IC,F}$ is set equal to the initial condition vector for the final orbit specified in Table 3. Similarly, the mass continuity constraints for the low-thrust nodes are defined as $\Delta m_i = m_{i,t+\Delta t} - m_{i+1,t}$ while the thrust direction vector constraints for the low-thrust nodes are written as $U_i = u_{i,x}^2 +$

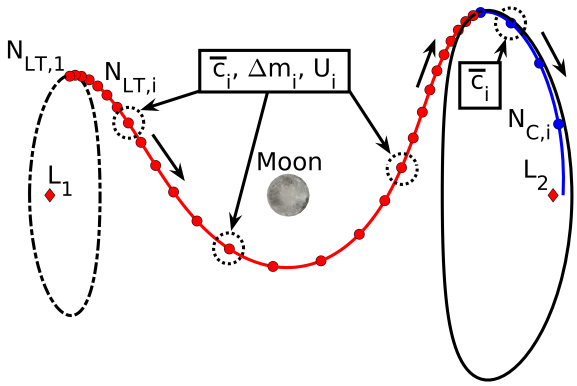


Figure 3: Free variable vector and constraints consisting of low-thrust and coast segments.

$u_{i,y}^2 + u_{i,z}^2 - 1$. Then, the entire constraint vector is written using these components as $\bar{F}(\bar{X}) = [\bar{c}_1, \ldots \bar{c}_F, \Delta m_1, \ldots, \Delta m_n, U_1, \ldots, U_n]^T$. Figure 3 depicts a free variable vector constructed using the results from MRPPO where spherical markers denote the initial state of a node, red arcs correspond to low-thrust nodes, blue arcs represent coast nodes, and constraints are noted between each node connection. These constraints are used to ensure that the solution recovered by *fmincon* is continuous in the state, mass, and thrust directions to within a specified tolerance.

Initial Guess

Each initial guess for the free variable vector used in *fmincon* is formulated using a reference trajectory developed by MRPPO. The low-thrust nodes are straightforwardly converted from each moving reference trajectory into low-thrust nodes for the corresponding initial guess. For the final orbit coast nodes, the final low-thrust node is naturally propagated forwards in time until the trajectory intersects the x-axis. This natural trajectory is then discretized into a specified number of coast nodes to form the final segment of the initial guess supplied to the optimization scheme in *fmincon*.

Objective Function

Once the initial guesses have been constructed, they are input to *fmincon* to generate locally optimal trajectories. However, *fmincon* requires that both the constraints and objective function be continuous and possess continuous first-order derivatives.[51] While the constraint vector formulation satisfies these requirements, the reward function defined for this trajectory design scenario is discontinuous and violates the requirements of *fmincon*. Thus, a distinct but comparable objective function is specified for use in *fmincon* to maximize the final mass of the spacecraft as $J = -m_{n,t+\Delta t}$ for a fixed flight time. Then, the recovered solutions necessarily are part of a distinct multi-objective solution space compared to the solutions recovered by MRPPO.

Optimization Method

Finally, *fmincon* possesses multiple optimization techniques that may be used to recover locally optimal solutions.[51]In this paper, Sequential Quadratic Programming (SQP) is used due to the demonstrated effectiveness of this method in a variety of trajectory design scenarios.[1,52,53] SQP methods formulate the optimization problem as a quadratic programming problem whereby solutions are iteratively updated until an optimal solution is found.[54] Additionally, SQP algorithms may

handle nonlinear constraints and do not require the initial guess to be feasible prior to generating an optimal solution.[54]

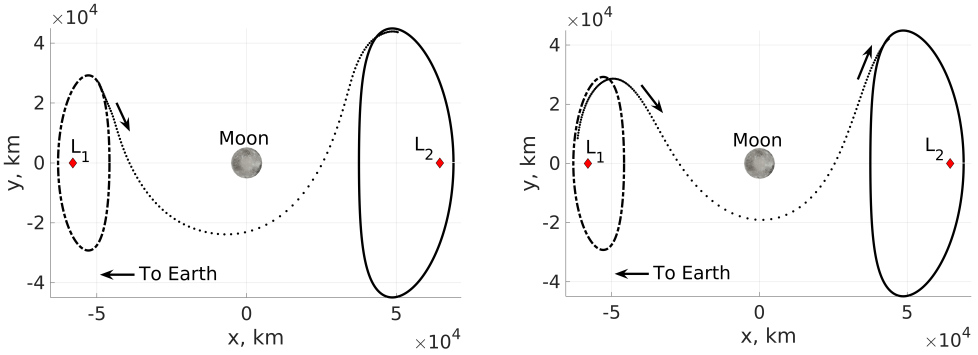
RESULTS

MRPPO with the moving reference modification is used to train four policies that recover low-thrust transfers between halo orbits in the Earth-Moon system. The four policies are assigned distinct reward functions to encourage the policies to uncover a distinct solutions within the multi-objective solution space. Once four policies are trained, the reference trajectories associated with each policy are used to construct initial guesses that are subsequently input to the optimization problem solved via *fmincon*. The resulting solution spaces from both methodologies are used to validate the results produced by MRPPO with the moving reference modification.

Transfers Recovered via MRPPO

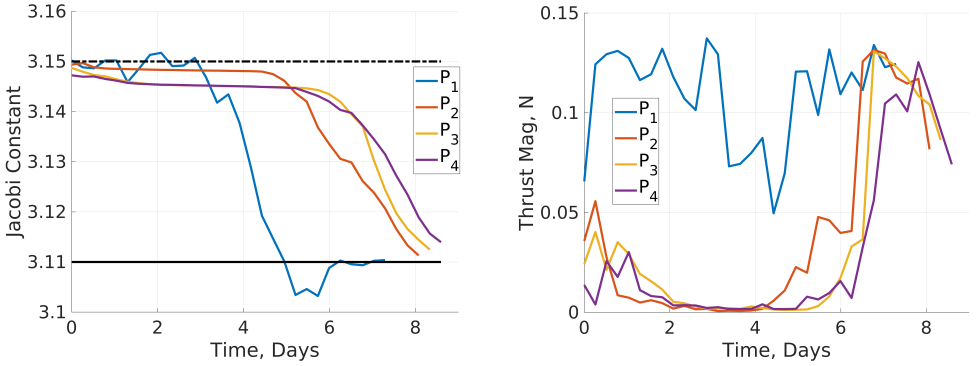
Four policies are trained using MRPPO with the moving reference modification to construct transfers for low-thrust-enabled SmallSats between halo orbits in the Earth-Moon system. Specifically, policy P_1 has no penalty on propellant mass usage while policy P_4 is assigned the highest penalty on propellant mass usage to encourage a variety of solutions within the multi-objective solution space. Once trained, both the policies and resulting reference trajectories for each policy are evaluated to uncover a subset of solutions within the propellant mass usage and flight time trade space. Figures 4a and 4b depict the reference trajectories developed by Policies 1 and 4 respectively, where the initial orbit is denoted by a black dashed arc, the final orbit by a solid black arc, and the reference trajectories are represented by dotted black arcs. While the trajectories may possess similar geometries, Figs. 5a and 5b demonstrate that the Jacobi constant and thrust magnitude profiles of each reference trajectory differ reflecting increases in the propellant mass usage penalty. Specifically, policy P_1 produces a higher thrust magnitude throughout the beginning of the transfer corresponding to an increase in the energy of the spacecraft. Conversely, policy P_4 produces negligible thrust and coasting until the spacecraft approaches the vicinity of the final periodic orbit. Policies P_2 and P_3 exhibit intermediary behavior with smaller levels of thrust than P_1 but larger than P_4. Additionally, since the policies are only trained to enter within a 5×10^{-3} hypersphere in position and velocity around the final orbit, a small difference in the Jacobi constant is evident. These reference trajectories are selected from the best trajectories generated during the training phase of the policies and serve as guides for the policies to construct transfers between the orbits.

The trained policies are evaluated on a common set of 1,000 perturbed initial conditions to facilitate direct comparisons between the policies. Figures 6a and 6b depict the 1,000 evaluation trajectories constructed by policies P_1 and P_4 respectively, and shaded in a variety of hues to aid in differentiation between the transfers. Both policies produce transfers that resemble the generated reference trajectories prior to converging on the final orbit. Additionally, the propellant mass usage and flight time is calculated for each evaluation trajectory. Figure 7a portrays the propellant mass usage and flight time values for each evaluation trajectory as circles and reference trajectory as diamonds constructed by each policy. The resulting trade space demonstrates a clear trend across the policies whereby propellant mass usage decreases at the expense of an increase in flight time as the propellant mass usage penalty increases. Similarly, Fig. 7b focuses on the propellant mass usage and flight time for each reference trajectory mirroring the trend in the evaluation trajectories. The reference trajectories for each policy accordingly correspond to the best trajectories generated and serve as a valuable basis for the initial guesses used in the optimization scheme.



(a) Policy 1 (b) Policy 4

Figure 4: Reference trajectories developed using MRPPO with the moving reference modification.

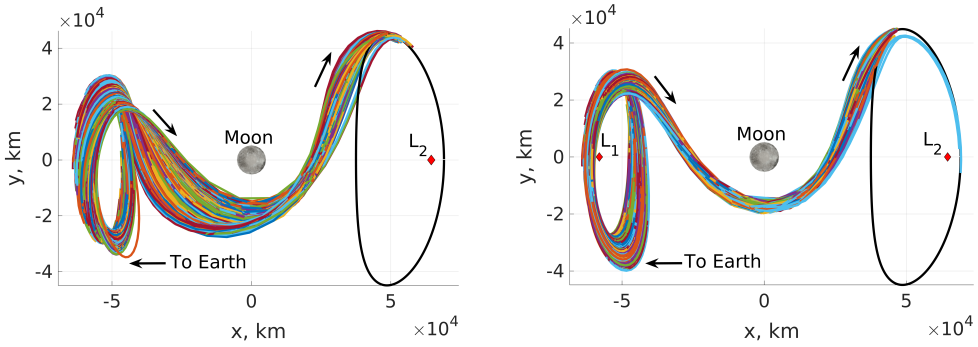


(a) Jacobi Constant (b) Thrust Magnitude

Figure 5: Characteristics of the reference trajectories generated by each policy during training.

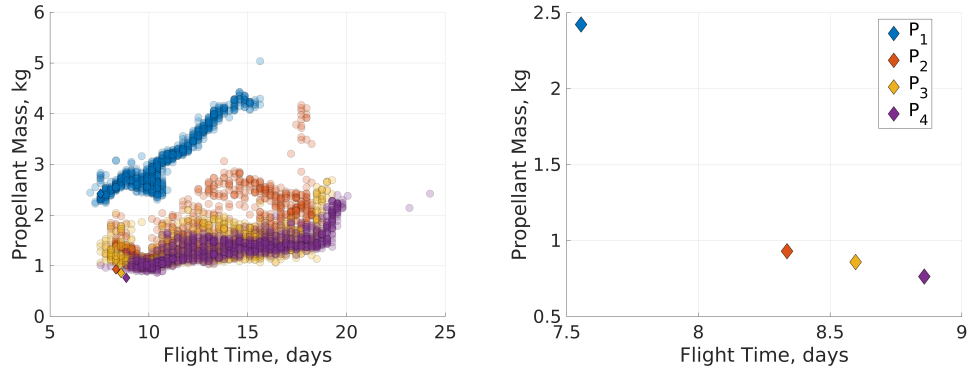
Transfers Recovered via Classical Optimization

The results autonomously generated by MRPPO with the moving reference modification are used to construct initial guesses for optimization in *fmincon* . Recall, the objective function for this optimization scheme is not equivalent to but only similar to the reward function used in MRPPO to train the policies. Then, *fmincon* with SQP is used to recover a distinct solution space. Figure 8 displays the propellant mass usage and flight time trade spaces recovered by SQP using the initial guesses constructed by MRPPO with the moving reference modification. In the figure, MR denotes the solutions recovered by MRPPO with the moving reference modification while SQP represents solutions recovered using *fmincon* with SQP. SQP recovers transfers with lower propellant mass usage than the transfers recovered by MRPPO. However, except for policy P_1 which has no incentive to decrease propellant mass usage, the difference in propellant mass usage is minimal and decreases as the flight time increases. The differences in propellant mass usage may be attributed to the differences in implementations: (1) the methodologies are recovering distinct but related solution spaces and (2) MRPPO is only required to enter the vicinity of the periodic orbit while *fmincon* must target it to within a nondimensional tolerance of 1E-10. Both approaches develop related solution spaces and demonstrate that as the flight time increases, the required propellant masses for both local minima approach similar values.



(a) Policy P_1 (b) Policy P_4

Figure 6: Evaluation trajectories generated by policies trained using MRPPO with the moving reference modification.



(a) Evaluation and reference trajectories (b) Reference trajectories

Figure 7: Propellant mass usage and flight time for the evaluation trajectories and reference trajectories generated by policies trained using MRPPO with the moving reference modification.

Examining the solutions produced by both algorithms enables the validation of the results produced by MRPPO with the moving reference modification. Figure 9 depicts two transfers associated with policy P_1: (a) the initial guess constructed using the policies trained by MRPPO and (b) the optimized transfer using SQP. In these figures, the red arcs correspond to low-thrust nodes while the blue arcs denote coast nodes. The geometry of the two transfers closely resemble one another with small differences in the solutions produced by *fmincon* due to the differences in the implementations. Additionally, Fig. 10 illustrates the two transfers associated with policy P_4 which is trained with a higher penalty on propellant mass usage. Both optimized transfers possess geometries that closely resemble the initial guesses developed by MRPPO with the moving reference modification. This demonstrates the benefit of using MRPPO with the moving reference modification to construct initial guesses whose geometries are approximately retained when input to an optimization scheme with a related objective function. Further, Figs. 11 and 12 depict the thrust magnitude and direction profiles associated with the MRPPO and *fmincon* transfers for policies P_1 and P_4, respectively. In these figures, the thrust magnitude profile associated with P_1 is altered by *fmincon* to resemble a bang-bang like profile with additional changes to the thrust direction. These changes may be due

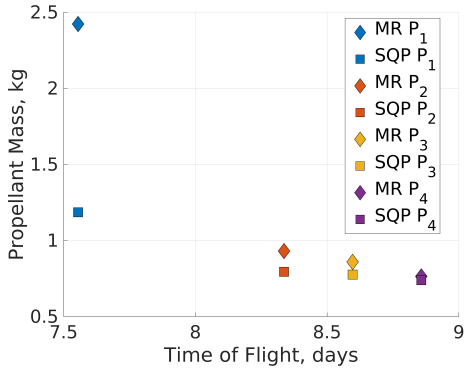
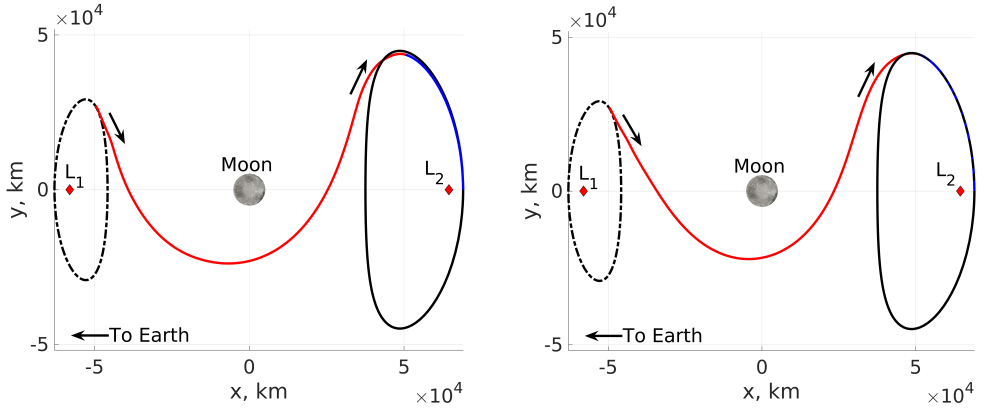


Figure 8: Solution spaces via optimization recovered from initial guesses developed by MRPPO with the moving reference modification.



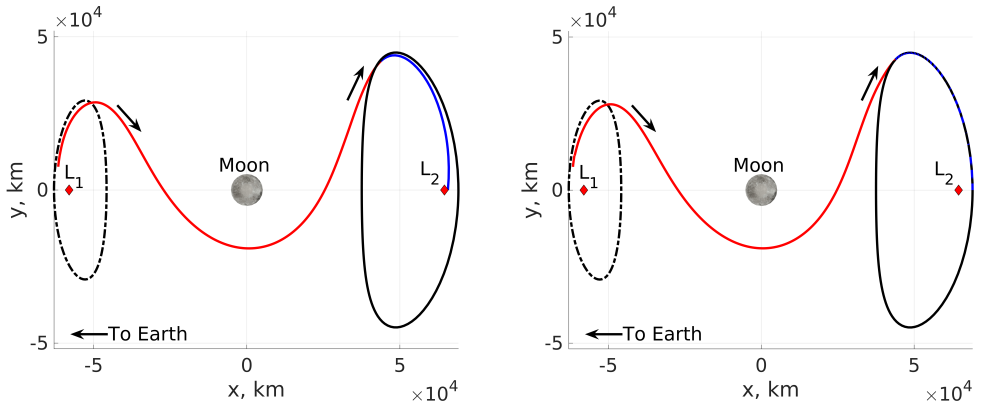
(a) Initial Guess (b) Sequential Quadratic Programming Transfer

Figure 9: Low-thrust transfers associated with policy P_1 from an L_1 northern halo orbit to an L_2 southern halo orbit in the Earth-Moon system.

to the differences in the objective functions between the two methodologies. However, the thrust profile for policy P_4 is negligibly altered by *fmincon*. The results produced by the optimization scheme suggests that the results generated by MRPPO with the moving reference modification lie close to the local optima in the multi-objective solution space. While SQP may recover transfers with a lower propellant mass usage for transfers with lower flight times, the difference in propellant mass usage decreases as the flight time increases. These results validate the results produced by MRPPO by demonstrating that the recovered solution space closely resembles a related solution space uncovered using a traditional optimization scheme.

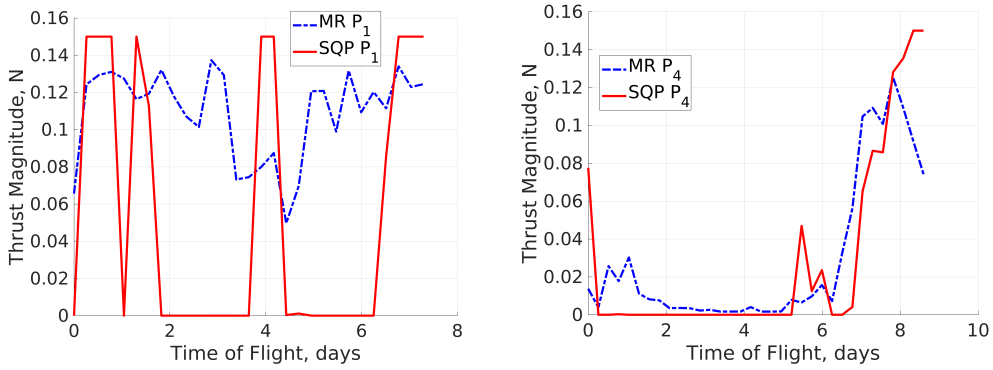
CONCLUSION

In this paper, the moving reference modification is incorporated into the MRPPO framework to enable policies to construct low-thrust transfers in a multi-body system without an initial guess from a human trajectory designer. Policies are trained to guide a low-thrust-enabled SmallSat from an L_1 northern halo orbit to an L_2 southern halo orbit in the Earth-Moon system without an initial guess and subject to distinct objective prioritizations. By developing low-thrust transfers in a chaotic



(a) Initial Guess (b) Sequential Quadratic Programming Transfer

Figure 10: Low-thrust transfers associated with policy P_4 from an L_1 northern halo orbit to an L_2 southern halo orbit in the Earth-Moon system.

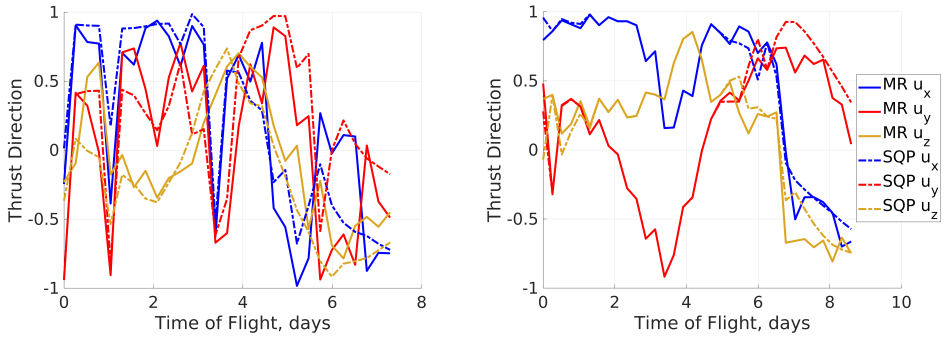


(a) Policy P_1 (b) Policy P_4

Figure 11: Thrust magnitude profiles for low-thrust transfers from an L_1 northern halo orbit to an L_2 southern halo orbit in the Earth-Moon system.

multi-body environment without a predefined reference trajectory, the time and effort required of a human trajectory designer is greatly reduced. Additionally, the policies trained by MRPPO uncover locally optimal solutions within the multi-objective solution space, specifically focusing on the flight time and propellant mass usage trade space.

The reference trajectories generated by MRPPO with the moving reference modification are leveraged as initial guesses in an optimization algorithm to validate the solution space produced by MRPPO. The trajectories are optimized using MATLAB's *fmincon* to solely maximize the final spacecraft mass, a distinct multi-objective solution space from that recovered by MRPPO with the moving reference modification. The recovered solution spaces from both approaches demonstrate comparable properties including similar propellant mass usages for a defined flight time. The results generated by *fmincon* with SQP confirm and validate the solution space produced by MRPPO with the moving reference modification while also evidencing the benefits of using MRPPO to construct low-thrust transfers in multi-body systems.



(a) Policy P_1 (b) Policy P_4

Figure 12: Thrust direction profiles in the Earth-Moon rotating frame for low-thrust transfers from an L_1 northern halo orbit to an L_2 southern halo orbit.

ACKNOWLEDGEMENTS

This work was supported by a NASA Space Technology Research Fellowship. Part of this research was performed at the University of Colorado Boulder. Part of this research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004).

REFERENCES

[1] C. J. Sullivan, J. Stuart, R. L. Anderson, and N. Bosanac, "Designing Low-Thrust Transfers to High-Inclination Science Orbits via Hybrid Optimization," *Journal of Spacecraft and Rockets*, 0, pp. 1–13. https://doi.org/10.2514/1.A34980.

[2] P. de Sousa-Silva, M. Terra, C. Mcinnes, and M. Ceriotti, "A Heuristic Strategy to Compute Ensembles of Trajectories for 3D Low-Cost Earth-Moon Transfers," *67th International Astronautical Congress, Guadalajara, MEX*, 09 2016.

[3] R. E. Pritchett, K. C. Howell, and D. J. Grebow, "Low-Thrust Transfer Design Based on Collocation Techniques: Applications in the Restricted Three-Body Problem," *2017 AAS/AIAA Astrodynamics Specialist Conference, Stevenson, WA*, 2017.

[4] I. Elliott, C. Sullivan, N. Bosanac, J. Stuart, and F. Alibay, "Designing Low-Thrust Trajectories for a SmallSat Mission to Sun–Earth L5," *Journal of Guidance, Control, and Dynamics*, Vol. 43, No. 10, 2020, pp. 1854–1864. https://doi.org/10.2514/1.G004993.

[5] S. Boone and J. McMahon, "Orbital Guidance Using Higher-Order State Transition Tensors," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 3, 2021, pp. 493–504. https://doi.org/10.2514/1.G005493.

[6] C. J. Sullivan, I. Elliott, N. Bosanac, F. Alibay, and J. R. Stuart, "Exploring the Low-Thrust Trajectory Design Space for SmallSat Missions to the Sun-Earth Triangular Equilibrium Points," *29th AAS/AIAA Space Flight Mechanics Meeting*, 2019.

[7] 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. https://doi.org/10.1016/j.actaastro.2019.04.037.

[8] A. Scorsoglio, R. Furfaro, R. Linares, and M. Massari, "Actor-Critic Reinforcement Learning Approach to Relative Motion Guidance in Near-Rectilinear Orbit," *29th AAS/AIAA Space Flight Mechanics Meeting, Ka'anapali, HI*, 2019.

[9] C. J. Sullivan and N. Bosanac, "Using Multi-Objective Deep Reinforcement Learning to Uncover a Pareto Front in Multi-Body Trajectory Design," *AAS/AIAA Astrodynamics Specialist Conference, Virtual*, 2020.

[10] C. J. Sullivan, N. Bosanac, R. L. Anderson, A. K. Mashiku, and J. R. Stuart, "Exploring Transfers between Earth-Moon Halo Orbits via Multi-Objective Reinforcement Learning," *IEEE Aerospace Conference, Virtual*, 2020. https://doi.org/10.1109/AERO50100.2021.9438267.

[11] S. Bonasera, I. Elliott, C. J. Sullivan, N. Bosanac, N. Ahmed, and J. McMahon, "Designing Impulsive Station-Keeping Maneuvers near a Sun-Earth L2 Halo Orbit via Reinforcement Learning," *31st AAS/AIAA Space Flight Mechanics Meeting, Virtual*, 02 2021.

[12] C. J. Sullivan and N. Bosanac, "Using Reinforcement Learning to Design a Low-Thrust Approach into a Periodic Orbit in a Multi-Body System," *30th AIAA/AAS Space Flight Mechanics Meeting, Orlando, FL*, 2020. https://doi.org/10.2514/6.2020-1914.

[13] 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, 2021, pp. 1–23. https://doi.org/10.1016/j.actaastro.2021.05.014.

[14] D. Miller and R. Linares, "Low-Thrust Optimal Control via Reinforcement Learning," *29th AAS/AIAA Space Flight Mechanics Meeting, Ka'anapali, HI*, 2019.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[16] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep Reinforcement Learning that Matters," *Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA*, 2018.

[17] T. Trinh, D. Vu, and M. Kimura, "Point-of-Conflict Prediction for Pedestrian Path-Planning," *Proceedings of the 12th International Conference on Computer Modeling and Simulation*, 2020, pp. 88–92.

[18] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a Quadrotor with Reinforcement Learning," *IEEE Robotics and Automation Letters*, Vol. 2, No. 4, 2017, pp. 2096–2103.

[19] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous Drone Racing with Deep Reinforcement Learning," *arXiv preprint arXiv:2103.08624*, 2021.

[20] V. Spandan, P. Bharadwaj, M. Bassenne, and L. Jofre, "Scalar Source Tracking in Turbulent Environments Using Deep Reinforcement Learning," *Proceedings of the 2018 Summer Program*, Center for Turbulence Research, 2018, pp. 155–164.

[21] W. S. Koon, M. W. Lo, J. E. Marsden, and S. D. Ross, *Dynamical Systems, The Three-Body Problem and Space Mission Design*. Marsden Books, 2006, pp. 8–11.

[22] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*. El Segundo, CA: Microcosm Press, 4th ed., 2013, pp. 1041–1042.

[23] D. C. Davis, S. M. Phillips, K. C. Howell, S. Vutukuri, and B. P. McCarthy, "Stationkeeping and Transfer Trajectory Design for Spacecraft in Cislunar Space," *AAS/AIAA Astrodynamics Specialist Conference, Stevenson, WA*, 2017.

[24] A. Zavoli and L. Federici, "Reinforcement Learning for Robust Trajectory Design of Interplanetary Missions," *Journal of Guidance, Control, and Dynamics*, Vol. 0, No. 0, 0, pp. 1–14. https://doi.org/10.2514/1.G005794.

[25] A. Harris, T. Teil, and H. Schaub, "Spacecraft Decision-Making Autonomy Using Deep Reinforcement Learning," *29th AAS/AIAA Space Flight Mechanics Meeting, Ka'anapali, HI*, 2019.

[26] M. Shirobokov, S. Trofimov, and M. Ovchinnikov, "Survey of Machine Learning Techniques in Spacecraft Control Design," *Acta Astronautica*, Vol. 186, 2021, pp. 87–97. https://doi.org/10.1016/j.actaastro.2021.05.018.

[27] D. Izzo, M. Märtens, and B. Pan, "A Survey on Artificial Intelligence Trends in Spacecraft Guidance Dynamics and Control," *Astrodynamics*, Vol. 3, No. 4, 2019, pp. 287–299. https://doi.org/10.1007/s42064-018-0053-6.

[28] 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. https://doi.org/10.1016/j.actaastro.2020.02.036.

[29] H. Holt, R. Armellin, N. Baresi, Y. Hashida, A. Turconi, A. Scorsoglio, and R. Furfaro, "Optimal Q-Laws via Reinforcement Learning with Guaranteed Stability," *Acta Astronautica*, 2021. https://doi.org/10.1016/j.actaastro.2021.07.010.

[30] L. Federici, B. Benedikter, and A. Zavoli, "Machine Learning Techniques for Autonomous Spacecraft Guidance during Proximity Operations," *AIAA Scitech 2021 Forum*, 2021. https://doi.org/10.2514/6.2021-0668.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-Level Control through Deep Reinforcement Learning," *Nature*, Vol. 518, 02 2015, pp. 529–533. https://doi.org/10.1038/nature14236.

[32] K. Hornik, M. Stinchcombe, H. White, *et al.*, "Multilayer Feedforward Networks are Universal Approximators.," *Neural Networks*, Vol. 2, No. 5, 1989, pp. 359–366. https://doi.org/10.1016/0893-6080(89)90020-8.

[33] G. Swirszcz, W. M. Czarnecki, and R. Pascanu, "Local Minima in Training of Deep Networks," *arXiv preprint arXiv:1611.06310*, 2016.

[34] P. Baldi and K. Hornik, "Neural Networks and Principal Component Analysis: Learning from Examples without Local Minima," *Neural networks*, Vol. 2, No. 1, 1989, pp. 53–58. https://doi.org/10.1016/0893-6080(89)90014-2.

[35] A. Y. Ng, D. Harada, and S. J. Russell, "Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping," *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 1999, p. 278–287.

[36] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-Dimensional Continuous Control Using Generalized Advantage Estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[37] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[38] M. Minsky, "Steps toward Artificial Intelligence," *in Proceedings of the IRE*, Vol. 49, No. 1, 1961, pp. 8–30. https://doi.org/10.1109/JRPROC.1961.287775.

[39] A. G. Barto, "Reinforcement Learning and Dynamic Programming," *6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems, Cambridge, MA*, Vol. 28, No. 15, 1995, pp. 407–412. https://doi.org/10.1016/S1474-6670(17)45266-9.

[40] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe1, J. Nham, N. Kalchbrenner, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, Vol. 529, No. 7587, 2016, pp. 484–489. https://doi.org/10.1038/nature16961.

[41] V. R. Konda and J. N. Tsitsiklis, "On Actor-Critic Algorithms," *Society for Industrial and Applied Mathematics Journal on Control and Optimization*, Vol. 42, April 2003, p. 1143–1166. https://doi.org/10.1137/S0363012901385691.

[42] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust Region Policy Optimization," *International Conference on Machine Learning, Lille, FRA*, 2015.

[43] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem, "What Matters In On-Policy Reinforcement Learning? A Large-Scale Empirical Study," *arXiv preprint arXiv:2006.05990*, 2020.

[44] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," *International Conference on Learning Representations*, 2019.

[45] Z. Zhang, X. Luo, T. Liu, S. Xie, J. Wang, W. Wang, Y. Li, and Y. Peng, "Proximal Policy Optimization with Mixed Distributed Training," *IEEE 31st International Conference on Tools with Artificial Intelligence*, 2019. https://doi.org/10.1109/ICTAI.2019.00206.

[46] M. S. Holubar and M. A. Wiering, "Continuous-Action Reinforcement Learning for Playing Racing Games: Comparing SPG to PPO," *arXiv preprint arXiv:2001.05270*, 2020.

[47] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

[48] R. L. Anderson and M. W. Lo, "Role of Invariant Manifolds in Low-Thrust Trajectory Design," *Journal of Guidance, Control, and Dynamics*, Vol. 32, November-December 2009, pp. 1921–1930. https://doi.org/10.2514/1.37516.

[49] N. Bosanac, F. Alibay, and J. R. Stuart, "A Low-Thrust Enabled SmallSat Heliophysics Mission to Sun-Earth L5," *IEEE Aerospace Conference, Big Sky, MT*, 2018.

[50] I. Levchenko, S. Xu, G. Teel, D. Mariotti, M. Walker, and M. Keidar, "Recent Progress and Perspectives of Space Electric Propulsion Systems Based on Smart Nanomaterials," *Nature communications*, Vol. 9, No. 1, 2018, pp. 1–19. https://doi.org/10.1038/s41467-017-02269-7.

[51] "MATLAB Optimization Toolbox Version 8.2," 2018. The MathWorks, Natick, MA, USA.

[52] J. T. Betts, "Very Low-Thrust Trajectory Optimization Using a Direct SQP Method," *Journal of Computational and Applied Mathematics*, Vol. 120, No. 1, 2000, pp. 27–40. https://doi.org/10.1016/S0377-0427(00)00301-0.

[53] N. L. O. Parrish, *Low Thrust Trajectory Optimization in Cislunar and Translunar Space*. PhD thesis, University of Colorado at Boulder, 2018.

[54] P. T. Boggs and J. W. Tolle, "Sequential Quadratic Programming," *Acta Numerica*, Vol. 4, 1995, pp. 1–51. https://doi.org/10.1017/S0962492900002518.