

LONG-TERM SPACECRAFT TRAJECTORY PREDICTION USING BEHAVIORAL MOTION PRIMITIVES

Austin Bodin^{*}, Natasha Bosanac[†], and Cole Gillespie[‡]

This paper presents a new approach to spacecraft trajectory prediction that uses behavioral motion primitives. A behavioral motion primitive summarizes a group of trajectories with a similar geometry and is labeled by their behaviors (e.g., maneuvering objective and intent) and spacecraft parameters. Each group of trajectories spans a primitive's region of existence, which is approximated using voxels. Uncertain state estimates are projected onto these regions of existence for short-term predictions. Sequences of composable primitives are then used to generate digestible long-term trajectory predictions. This procedure is demonstrated in the Earth-Moon circular restricted three-body problem.

INTRODUCTION

An increased number of future missions are expected to support scientific exploration, long-term infrastructure, and defense applications in cislunar space. The associated spacecraft may possess diverse behaviors, i.e., intents and control profiles, as well as properties such as propulsion system and mass. The range of possible behaviors and properties produces a diverse array of motions that a spacecraft could follow within the chaotic dynamical environment of cislunar space. Furthermore, similar trajectory geometries can be generated through distinct combinations of spacecraft behaviors and properties. As a result, with limited or no information about the behaviors and capabilities of an observed spacecraft, trajectory prediction from uncertain state estimates becomes a challenging task that can produce an overwhelming amount of information for an operator or analyst.

Initial orbit determination (IOD) is a critical task for space operations. Foundational methods such as Laplace's method, Gauss' method, Double r-iteration, and Gooding's method^{1,2} construct orbit fits that often rely on mathematical relationships from the two-body problem, which can lead to poor approximations in a multi-body system. To refine the state estimate and provide state uncertainty, large numbers of these measurements are often used in a batch estimation method such as weighted nonlinear least squares.³ The IOD estimate is then used to predict possible trajectories of the spacecraft. An existing technique is Monte Carlo analysis, which relies on repeated random sampling to numerically generate trajectories,⁴ potentially imposing a high computational burden when there is limited information on the behavior and properties of the spacecraft as well as producing a large amount of information. These heritage methods have proven valuable for IOD in

^{*}Graduate Research Assistant, Ann & H.J. Smead Department of Aerospace Engineering Sciences, Colorado Center for Astrodynamics Research, University of Colorado Boulder, Boulder, CO, 80303.

[†]Associate Professor, Ann & H.J. Smead Department of Aerospace Engineering Sciences, Colorado Center for Astrodynamics Research, University of Colorado Boulder, Boulder, CO, 80303.

[‡]Graduate Research Assistant, Ann & H.J. Smead Department of Aerospace Engineering Sciences, Colorado Center for Astrodynamics Research, University of Colorado Boulder, Boulder, CO, 80303.

dynamical environments where the gravity field of a single celestial body is dominant, but they have limited applicability in cislunar space.

Complex path and movement prediction problems have been encountered across a diverse array of disciplines. In robotics, the concept of motion primitives has been used to represent fundamental building blocks of motion.⁵ These motion primitives can be extracted from groups of similar motion within a set of observational or simulated data.^{6–8} A library of motion primitives then supplies a discrete and digestible summary of the solution space. By sequencing composable motion primitives, analogous to assembling building blocks with compatible connections, more complex paths can be rapidly generated.⁹ Due to their capability to summarize distinct types of paths or movements, motion primitives have been useful for supplying rapid and digestible path and behavioral predictions. For example, Habibi, Jaipuria, and How have used motion primitives to predict future pedestrian motions.¹⁰ Edelhoff et al. also used motion primitives to extract behavioral models from animal movement observations and identify behavioral changes.¹¹

Motion primitives have been used for spacecraft trajectory design in multi-body gravitational systems by Smith and Bosanac to summarize a set of arcs with a similar geometry in the circular restricted three-body problem (CR3BP).¹² These groups of geometrically similar arcs have been extracted using clustering to separate motions with distinct geometries, with substantial improvements recently developed by Bosanac¹³ as well as Gillespie, Miceli, and Bosanac.¹⁴ Smith and Bosanac also defined a graph-based procedure to identify sequences of motion primitives that produce geometrically distinct initial guesses for spacecraft trajectory design.¹² Miceli and Bosanac have extended this work to improve the graph construction process, incorporate constraints, and improve the graph search algorithm to produce a more diverse array of primitive sequences.^{15,16} Leveraging this foundation, Gillespie, Miceli, and Bosanac introduced behavioral motion primitives to encode the array of intents, control profiles, and spacecraft parameters that produce each group of geometrically similar arcs for natural and thrust-enabled motions in cislunar space.¹⁴

This paper focuses on using behavioral motion primitives for long-term spacecraft trajectory prediction in the Earth-Moon CR3BP. First, this paper leverages the diverse library of behavioral motion primitives generated by Gillespie, Miceli, and Bosanac.¹⁴ Then, the region of the phase space encompassed by arcs that geometrically resemble each primitive, labeled its region of existence, is approximated using voxels. Initial state estimates are projected onto the library of behavioral motion primitives. Any primitives with regions of existence that overlap with an uncertainty hyperellipsoid centered on the state estimate supply short-term motion predictions. Next, recent work by Miceli and Bosanac¹⁵ is extended to construct a graph of behavioral motion primitives. This graph captures the connections between primitives that are sequentially composable in the configuration space, estimated via intersections of the voxel approximation of their regions of existence. Searching this graph produces primitive sequences that are used to rapidly generate a digestible summary of long-term possible trajectory predictions. This new approach to trajectory prediction is demonstrated by exploring possible long-term future motions of a spacecraft from an uncertain state estimate in the lunar vicinity and in the planar Earth-Moon CR3BP, assuming only impulsive maneuvers.

BACKGROUND

Dynamical Model

The CR3BP is used to approximate the dynamical environment governing the motion of a spacecraft. This model assumes that the mass of the spacecraft is negligible compared to the two primary

bodies, the Earth and the Moon, which follow circular orbits.¹⁷ The primary bodies are also modeled with the same gravity fields as point masses with a constant mass.¹⁷

State vectors are specified in nondimensional units using a rotating frame. The rotating frame is defined with an origin at the center of mass of the Earth-Moon system. The axes $\{\hat{x}, \hat{y}, \hat{z}\}$ are defined with $\hat{x}$ directed from the Earth to the Moon, $\hat{z}$ aligned with the orbital angular momentum vector of the primaries, and $\hat{y}$ completing the right-handed, orthogonal triad.¹⁷ Quantities are also nondimensionalized using characteristic parameters for length (l^*), mass (m^*), and time (t^*): $l^* = 384,400$ km is the distance between the Earth and Moon, $m^* = 6.046 \times 10^{24}$ kg is the total system mass, and $t^* = 3.752 \times 10^5$ s produces a nondimensional period of the primaries equal to 2π .¹⁸

The equations of motion for the spacecraft are typically expressed in the Earth-Moon rotating frame. The nondimensional state of the spacecraft is first defined as $\mathbf{x} = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^T$. Then, the equations of motion are written as¹⁷

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

where $U^* = 0.5(x^2 + y^2) + (1 - \mu)/r_1 + \mu/r_2$, and $r_1 = \sqrt{(x + \mu)^2 + y^2 + z^2}$ and $r_2 = \sqrt{(x - 1 + \mu)^2 + y^2 + z^2}$ are the distances to the primaries. In the Earth-Moon system, the mass ratio of the primaries is $\mu = 0.01215058439470971$.¹⁸ The Jacobi constant is an integral of motion equal to $C_J = 2U^* - \dot{x}^2 - \dot{y}^2 - \dot{z}^2$.¹⁷ This energy-like quantity supplies insight into allowable regions of motion.¹⁷ In this dynamical model, the five libration points, L_1 - L_5 , are stationary solutions in the rotating frame.¹⁷ Periodic orbits are solutions that repeat after a finite time in the rotating frame. Finally, hyperbolic invariant manifolds asymptotically depart or approach a libration point, periodic orbit, or quasi-periodic orbit.¹⁷

Differential Geometry

Differential geometry supports the analysis of nonlinear trajectories. At an instant of time t , the position, velocity, and acceleration vectors are equal to $\mathbf{r}(t) = [x(t), y(t), z(t)]^T$, $\mathbf{v}(t) = [\dot{x}(t), \dot{y}(t), \dot{z}(t)]^T$, and $\mathbf{a}(t) = [\ddot{x}(t), \ddot{y}(t), \ddot{z}(t)]^T$, respectively. Over the time interval $t \in [t_0, t_f]$, a trajectory propagated from an initial state traverses a distance equal to the arclength s_p .¹⁹ In addition, the velocity hodograph traces out a curve of arclength s_v , labeled “velocity arclength,” equal to the cumulative change in velocity over the trajectory. These quantities are calculated as

$$s_p = \int_{t_0}^{t_f} \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} dt \quad \text{and} \quad s_v = \int_{t_0}^{t_f} \sqrt{\ddot{x}^2 + \ddot{y}^2 + \ddot{z}^2} dt. \quad (2)$$

At any location along the trajectory, the curvature captures the deviation of the path from a straight line. The unsigned curvature is calculated as follows,²⁰ with a singularity when the speed is zero:

$$\kappa(\mathbf{x}) = \frac{\sqrt{(\dot{x}\ddot{y} - \dot{y}\ddot{x})^2 + (\dot{z}\ddot{x} - \dot{x}\ddot{z})^2 + (\dot{y}\ddot{z} - \dot{z}\ddot{y})^2}}{(\dot{x}^2 + \dot{y}^2 + \dot{z}^2)^{3/2}}. \quad (3)$$

At maxima in curvature, the shape of the trajectory changes most rapidly. In the CR3BP, these maxima often occur close to apses relative to meaningful reference points.²¹

Density-Based Clustering

Clustering algorithms automatically discover groups of similar data while separating dissimilar data.²² In this paper, density-based techniques are used to identify regions in a finite-dimensional feature space where data points exhibit sufficient local concentration.²² Specifically, two algorithms are employed: Density-Based Spatial Clustering of Applications with Noise (DBSCAN)²³ and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN).²⁴

DBSCAN was introduced by Ester, Kriegel, Sander, and Xu to detect arbitrarily shaped clusters by linking points with overlapping, sufficiently dense neighborhoods.²³ A core point possesses at least m_{pts} neighbors within a radius of ϵ , labeled its ϵ -neighborhood. Those points that lie within the ϵ -neighborhood of a core point but do not themselves satisfy the core point criterion are labeled border points. The remaining points are labeled as noise. Clusters in DBSCAN consist of all core points that are density-connected, i.e., reachable through a chain of overlapping core neighborhoods, and their associated border points. This algorithm is useful when the governing parameters m_{pts} and ϵ can be selected heuristically. In this paper, MATLAB's *dbscan* function is employed.²⁵

HDBSCAN, developed by Campello, Moulavi, and Sander, extends DBSCAN by building a cluster hierarchy over varying density thresholds.²⁴ HDBSCAN begins by defining the mutual reachability distance between two points as the maximum of their pairwise distance and each point's core distance, i.e., the distance to its m_{pts} th nearest neighbor. The mutual reachability distance between all points is used to construct a minimum spanning tree, which captures the data's intrinsic density structure. This tree is then condensed into a cluster hierarchy where each node represents a candidate cluster containing at least $m_{min,clust}$ points. A stability criterion based on excess of mass is used to extract the most persistent clusters from this hierarchy, whereas points that fail to meet density requirements remain classified as noise.²⁴ HDBSCAN does not depend on a fixed neighborhood radius and can accommodate clusters of differing densities and sizes. HDBSCAN is primarily governed by two parameters: m_{pts} and $m_{min,clust}$.²⁴ An explicit reachability threshold ϵ_{merge} may also be applied to constrain cluster formation.²⁶ In this paper, HDBSCAN is implemented via the Python *hdbscan* library.²⁷

Numerically Correcting Trajectories

Multiple shooting is used to recover a continuous trajectory from a discontinuous initial guess.²⁸ In this paper, a free variable vector $\mathbf{V}$ is defined as

$$\mathbf{V} = \begin{bmatrix} \mathbf{x}_{1,0}^T & \Delta t_1 & \mathbf{x}_{2,0}^T & \Delta t_2 & \dots & \mathbf{x}_{n,0}^T & \Delta t_n \end{bmatrix}^T, \quad (4)$$

where $\mathbf{x}_{i,0}$ and Δt_i are the initial state and time of flight along the i th arc, respectively. A constraint vector $\mathbf{F}_0(\mathbf{V})$ that enforces continuity in the configuration space while allowing impulsive maneuvers between subsequent arcs is defined as

$$\mathbf{F}_0(\mathbf{V}) = \begin{bmatrix} \mathbf{r}_{1,f}^T - \mathbf{r}_{2,0}^T, \mathbf{r}_{2,f}^T - \mathbf{r}_{3,0}^T, \dots, \mathbf{r}_{n-1,f}^T - \mathbf{r}_{n,0}^T \end{bmatrix}^T, \quad (5)$$

where $\mathbf{r}_{i,0}$ is the initial position vector along the i th arc and $\mathbf{r}_{i,f}$ is the final position along the i th arc. Full state continuity can be used if a maneuver is not allowed. Additional boundary and path constraints can also be included as needed. Inequality constraints can be included by converting them to equality constraints and appending slack variables β_i to the free variable vector. These additional constraints are appended to $\mathbf{F}_0$ to form the full constraint vector $\mathbf{F}$. Then, an initial guess

for the free variable vector is iteratively updated via Newton’s method. These updates continue until $|\mathbf{F}(\mathbf{V})| < \tau$, where τ is a specified tolerance. At each iteration, the derivatives of the constraints with respect to the free variables are calculated analytically.

Voxel Representations

A voxel is a three-dimensional analog of a pixel often used for computer graphics and medical or scientific visual analysis.²⁹ Voxels are typically cubes with uniform size that are used to discretize more complex shapes. This volume representation supports efficient rendering, simple intersection or overlap calculations, and variable resolution. Their positions can be defined through a spatial data structure (e.g., an octree or k-d tree), or each voxel can carry its own absolute coordinates.²⁹ Voxels can also be encoded with information such as color or a selected parameter value.²⁹ In this paper, voxels are used to represent volumes of the configuration space with uniform side lengths s_i and specified center positions $\mathbf{c}_i$, as depicted in Figure 1. Note that a similar concept has been used by Howell et al.³⁰ to define cells that approximate the tube-like surfaces formed by stable and unstable manifolds in the CR3BP.

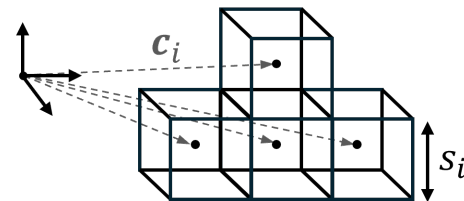


Figure 1: A group of voxels, each with a side length s_i and center positions $\mathbf{c}_i$.

Graph Search Algorithms

Two fundamental search algorithms are Dijkstra’s³¹ and A*,³² both of which are used in this work. These algorithms are used within weighted graphs with nonnegative edge weights to identify the shortest path from one node, the start node, to another node in the graph, the end node.^{31,32} The term “shortest path” is the path with the minimum sum of edge weights, and the edge weights in a graph are nonnegative values that describe the ease of moving from one node to another.³¹

The A* search algorithm identifies the shortest path between two nodes in fewer steps than Dijkstra’s algorithm due to the use of a heuristic.³² The A* algorithm evaluates each candidate node based on the sum of the cost from the start to that node, $g(n)$, and the heuristic cost from that node to the goal, $h(n)$, selecting the node with the lowest total $f(n) = g(n) + h(n)$ for expansion.³² The heuristic guides the search algorithm at each node by providing approximate knowledge of the true cost to the end node.³² The heuristic must be “admissible,” i.e., $h(n) \leq h^*(n)$, where $h^*(n)$ is the true cost from node n to the end node.³² As $h(n)$ approaches 0, A* explores more of the graph and effectively behaves like Dijkstra’s algorithm.³² As $h(n)$ approaches $h^*(n)$, A* explores fewer nodes, traversing only those necessary to reach the goal.

Yen’s algorithm identifies the K -shortest loop-less paths between a start node and an end node in a weighted graph with nonnegative edge weights.³³ This algorithm begins by computing the first shortest path using, for example, Dijkstra’s algorithm.³³ Additional candidate paths $k = 2, 3, \dots, K$ are generated using each node i in the $(k-1)$ th path.³³ First, the segment from the start node to node i is defined as a root path. Edges that would recreate any of the previous $(k-1)$ root paths up to i are temporarily removed from the graph to produce a subgraph.³³ Next, a spur path is computed from node i to the goal by searching the subgraph using, for example, Dijkstra’s algorithm.³³ Finally, the spur path is concatenated with the root path to form a full candidate path.³³ These candidate paths are stored and the lowest-cost candidate is selected as the next shortest path.³³ This spur-and-root procedure repeats until the top K loop-less paths have been identified.³³

TECHNICAL APPROACH

Step 1: Constructing a Behavioral Motion Primitive Library

A library of behavioral motion primitives is generated to discretely summarize a segment of the continuous solution space. This step closely follows the procedure previously developed by Gillespie, Miceli, and Bosanac.¹⁴ In this paper, the behavioral motion primitives are generated to summarize natural arcs along the stable and unstable manifolds of Lyapunov orbits near the Moon.

The stable and unstable manifolds of Lyapunov orbits are numerically approximated. First, selected orbits are discretized into 500 states that are equally distributed in arclength. These states are perturbed to produce initial states in the stable or unstable eigenspaces and then propagated for a duration of $\Delta t_{prop} = \Delta t_{pertdoub} + 3$ months in the CR3BP. $\Delta t_{pertdoub}$ is the perturbation doubling time, calculated from the unstable eigenvalue of the monodromy matrix.³⁴ Propagation terminates early upon impacting a primary body or exceeding a distance of one nondimensional unit from the Moon. During propagation, the curvature maxima along the trajectories are recorded.

Arcs are then sampled as “building-block” sized segments of each set of approach or departure trajectories.¹⁴ Each arc is defined to start at 1) the initial state of a trajectory for departure arcs, 2) a maximum in curvature, or 3) a termination condition for approach arcs. The arcs then encompass as many as four subsequent maxima in curvature before terminating. The final state along each arc is either 1) a maxima in curvature, 2) the initial state used to generate a stable manifold trajectory in backward time, or 3) a state that satisfies a termination condition along an unstable manifold.

Each arc is then sampled geometrically.¹⁴ First, the arc is sampled at its initial state, any maxima in the curvature, and its final state. In between these initial samples, two additional samples are spaced evenly in arclength. Figure 2 depicts a trajectory sampled in this manner where the red diamonds mark the initial samples and the black diamonds locate the intermediate samples. This sampling methodology ensures that samples are placed in geometrically meaningful regions. The region in between each sample is designated a “section.” For the example in Figure 2, the arc is divided into twelve sections, each numbered in the figure, between thirteen sampled states.

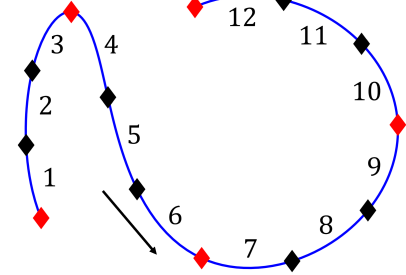


Figure 2: Trajectory discretized into twelve sections, encompassing five maxima in the curvature.

These sampled states are used to create two feature vectors that describe a continuous arc.^{14,35} A shape-based feature vector $\mathbf{f}_s$ and position-based feature vector $\mathbf{f}_p$ are defined as

$$\mathbf{f}_s = [\hat{\mathbf{v}}_1^T, \hat{\mathbf{v}}_2^T, \dots, \hat{\mathbf{v}}_n^T]^T \quad \mathbf{f}_p = [\mathbf{r}_1^T, \mathbf{r}_2^T, \dots, \mathbf{r}_n^T]^T, \quad (6)$$

where n is the number of sampled states, and $\hat{\mathbf{v}}_i$ is the velocity unit vector. Each feature vector possesses a dimension of $3n \times 1$ to summarize the geometry of the arc in the rotating frame.

Arcs sampled from the same fundamental solution are clustered by their geometry. First, the arcs are coarsely clustered in the shape-based feature vector space using HDBSCAN.^{24,27} The governing parameters are selected as $m_{pts} = 4$, $m_{min,clust} = 5$, and $\epsilon_{merge} = 2 \sin(5^\circ/2)$.³⁵ Each coarse group is then refined using a process developed by Bosanac that is modeled after convoy detection to define each cluster as a set of arcs with a consistently similar shape and path.^{35,36} First, the i th sampled state along each arc in the coarse group is described by $\mathbf{f}_s$ and $\mathbf{f}_p$, producing two three-dimensional feature vectors. These sampled states are clustered in each feature vector space

independently using DBSCAN.²³ The governing parameters are selected as $m_{pts} = 4$ and $\epsilon = (m_{pts} + 1) \max(e_{maxk}, \epsilon_{thresh})$, where e_{maxk} is the $m_{minclust}$ -largest distance from each member to its nearest neighbor and ϵ_{thresh} is a specified threshold, equal to $2 \sin(5^\circ/2)$ in $\mathbf{f}_s$ and 10^{-3} in $\mathbf{f}_p$.³⁵ This procedure is repeated for all n states along the arc to produce $2n$ clustering results. Any arcs that are consistently grouped together in all clustering results produce a refined cluster. This step produces a set of clusters $\mathcal{L}$, localized to each half-manifold of each periodic orbit.

Similar motion types that exist across different half-manifolds are aggregated using pairwise merging decisions. First, the nearest neighboring representatives are identified in each of $\mathbf{f}_s$ and $\mathbf{f}_p$. The arcs in each pair of candidate clusters are input to the cluster refinement process. If any members are grouped together, the two clusters are merged.^{14,35} This aggregation process produces a set of global clusters $\mathcal{G}_j$ summarizing arcs that approach or depart periodic orbits along one family.

Periodic orbits are discretely sampled along continuous families and their curvature profiles are analyzed to extract motion primitives. This procedure was previously presented by Gillespie, Miceli, and Bosanac.¹⁴ Groups of geometrically similar orbits within each family are generated from neighboring, discretely sampled members possessing the same number of curvature maxima.

To support trajectory prediction, each cluster is summarized by its motion primitive and region of existence.¹⁴ First, the motion primitive is selected as the medoid of the cluster evaluated in the position-based feature vector space. Next, the associated region of existence, i.e., the volume that the cluster encompasses within the phase space, is approximated through a discrete hypervoxel representation. To support the discretization, the continuous arcs are further sampled in even increments of 5×10^{-4} non-dimensional units in the arclength and velocity arclength. An example is depicted in Figure 3, where the states sampled along a cluster of geometrically similar arcs that depart an L_1 Lyapunov orbit are plotted in blue and the primitive arc is overlaid in black. Additionally, the red diamond locates L_1 , and the gray sphere represents the Moon, drawn to scale.

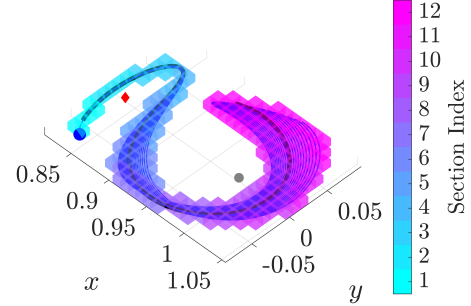


Figure 3: The region of existence for a motion primitive departing an L_1 Lyapunov orbit.

The region of existence associated with a motion primitive is approximated via a collection of position and velocity voxels to reduce data storage and complexity. First, a three-dimensional position voxel is defined as a cube-shaped region in the configuration space that contains at least one state along any arcs within the cluster associated with the primitive. The velocity vectors of the states that lie within each position voxel are then summarized by a set of three-dimensional velocity voxels that each encompass a cube-shaped region in $(\dot{x}, \dot{y}, \dot{z})$. These two sets of voxels are stored hierarchically: each velocity voxel is stored in a nested set associated with a specific position voxel. Each velocity voxel is also encoded with the arc and section index for each sample. To define these position and velocity voxels, their centers $\mathbf{c}_p$ and $\mathbf{c}_v$ are located at integer multiples of 0.01 nondimensional units in each state component. The position and velocity voxels are axis-aligned and possess side lengths of $s = 0.01$ nondimensional units. Figure 3 displays a motion primitive's region of existence, approximated by 185 position voxels that are projected onto the configuration space and colored by their highest section index.

This paper uses a library of 6,861 motion primitives. This library includes arcs that 1) follow, approach, or depart selected L_1 and L_2 Lyapunov orbits; and 2) the Moon-centered distant prograde

orbits, distant retrograde orbits, and low prograde orbits. Ongoing work includes incorporating spatial and thrusting primitives, as presented by Gillespie, Miceli, and Bosanac.¹⁴

Step 2: Estimation Mapping

An uncertain state estimate is projected onto the motion primitives in the library to identify possible short-term motion types and supply a digestible summary of short-term trajectory predictions. First, a 3σ uncertainty hyperellipsoid centered at an estimate mean $\bar{x}$ with covariance Σ is defined analytically in the phase space. Here, a 3σ uncertainty hyperellipsoid refers to the collection of points with a squared Mahalanobis distance equal to³⁷

$$d_M^2(\mathbf{x}) = (\mathbf{x} - \bar{\mathbf{x}})^T \Sigma^{-1} (\mathbf{x} - \bar{\mathbf{x}}) \leq 9 \quad (7)$$

and corresponding to a 82.64% confidence region in the six-dimensional phase space.

To support comparison to a six-dimensional state, the hierarchical collection of position and velocity voxels that summarize a region of existence are used to form a set of six-dimensional hypervoxels. Specifically, each three-dimensional velocity voxel nested within one of the three-dimensional position voxels associated with a primitive's region of existence are combined. The i th six-dimensional hypervoxel possesses a center located at $\mathbf{h}_i = [\mathbf{c}_{p,i}^T, \mathbf{c}_{v,i}^T]^T$ with a side length of $s = 0.01$ nondimensional units in all six state components. The hypervoxel also inherits the section and arc indices of the velocity voxel.

Identification of hypervoxels that intersect the uncertainty hyperellipsoid is performed in two steps. First, the squared Mahalanobis distance at each hypervoxel center, $d_M^2(\mathbf{h}_i)$, is evaluated. If $d_M^2(\mathbf{h}_i) \leq 9$, the hypervoxel intersects the uncertainty hyperellipsoid. Otherwise, a constrained root-finding procedure is used to search for a state $\mathbf{x}$ within the hypervoxel that lies on the hyperellipsoid surface. The procedure solves for an $\mathbf{x}$ such that $d_M^2(\mathbf{x}) = 9$, subject to constraints that $\mathbf{x}$ remains within the hypervoxel bounds. If a solution exists, then the corresponding hypervoxel is marked as intersecting. Intersecting hypervoxels are stored along with the section and arc indices. The process is repeated using the voxel representation of the region of existence associated with each primitive in the library to identify the subset that intersect the uncertainty hyperellipsoid.

In this paper, an example observation is generated through the random sampling of trajectories. A state along a trajectory that departs an L_1 Lyapunov orbit is propagated 1000 times for a duration of one day from t_0 to t_1 with a standard deviation of 20 m in position and 10 m/s in velocity. After propagating, a mean state estimate $\bar{x} = [0.9523, -0.0648, 0.0000, 0.2942, 0.0945, 0.0000]^T$ is extracted from the terminal states along with a covariance matrix $\Sigma \in \mathbb{R}^6$ that produces the standard deviations $\sigma_x = [0.89, 0.82, 0.63, 5.48, 4.85, 1.37] \times 10^{-2}$. The first three elements correspond to position components, and the last three to velocity components. These trajectories are visualized in Figure 4 in black with $\bar{x}$ as a blue star and the final time labeled t_1 . The resulting three-sigma uncertainty hyperellipsoid is projected onto the configuration space and plotted as a transparent red ellipsoid.

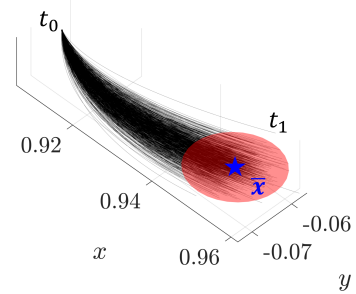


Figure 4: Trajectories propagated to generate an uncertainty hyperellipsoid and $\bar{x}$.

Comparing the state estimate depicted in Figure 4 to the library of 6,861 motion primitives produces a set of 18 intersecting motion primitives in approximately three minutes. In this paper, all

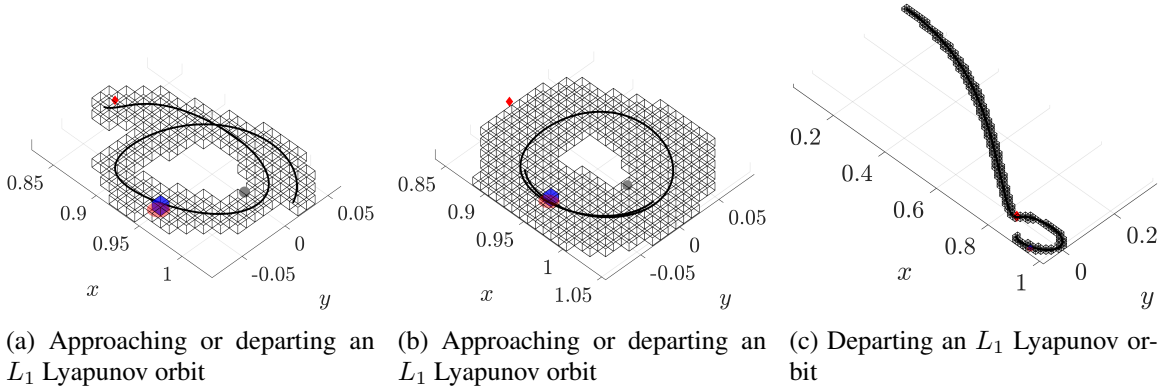


Figure 5: Three regions of existence that intersect with the uncertainty hyperellipsoid in Figure 4.

computations are performed using a 6-core AMD Ryzen 5 7600X CPU. The intersecting motion primitives all approach or depart an L_1 Lyapunov orbit naturally. Three of these primitives and their regions of existence are plotted in Figure 5, where intersecting position voxels are highlighted in blue and the uncertainty hyperellipsoid projected onto the configuration space is displayed in transparent red. The regions of existence all possess similar shapes near the initial state estimate, but the subsequent geometries and itineraries vary drastically: Figure 5(a) displays a self-intersecting geometry that gradually moves towards L_1 , Figure 5(b) depicts motion that remains within the vicinity of the Moon, and Figure 5(c) shows a third possible geometry that departs through the L_1 gateway. Thus, the state estimate and uncertainty level could lead to a spacecraft following a variety of itineraries.

Step 3: Motion Primitive Graph Construction

The sequential composability of all pairs of behavioral motion primitives from the library is calculated to support long-term trajectory prediction. Specifically, two motion primitives are labeled as sequentially composable if the position voxels representing their two regions of existence overlap in the configuration space. To demonstrate this definition, consider the regions of existence of two primitives plotted in Figure 6(a): region of existence 1 (red) summarizes trajectories departing an L_1 Lyapunov orbit whereas region of existence 2 (blue) summarizes trajectories approaching an L_2 Lyapunov orbit. The position voxels are colored in darker shades as the section index increases, supplying insight into the direction of motion along the associated trajectories. Their intersecting position voxels are colored purple. A motion primitive graph is generated to summarize the sequential composability of all pairs of primitives from the library. This step builds upon prior work by Miceli and Bosanac¹⁵ but increases the accuracy by directly using the voxel representations.

The motion primitive graph is first populated with nodes. Each node represents a single section along the region of existence of one motion primitive. Thus, a region of existence discretized into 12 sections contributes 12 nodes to the graph.

Zero-weight, directed edges connect sequential sections associated with the same motion primitive. Accordingly, these “flow” edges reflect that a spacecraft can naturally traverse the continuous sections of trajectories with a similar geometry in forward time. The final section node for periodic orbits is connected with a similar, zero-weight edge to the first node to encode periodicity.

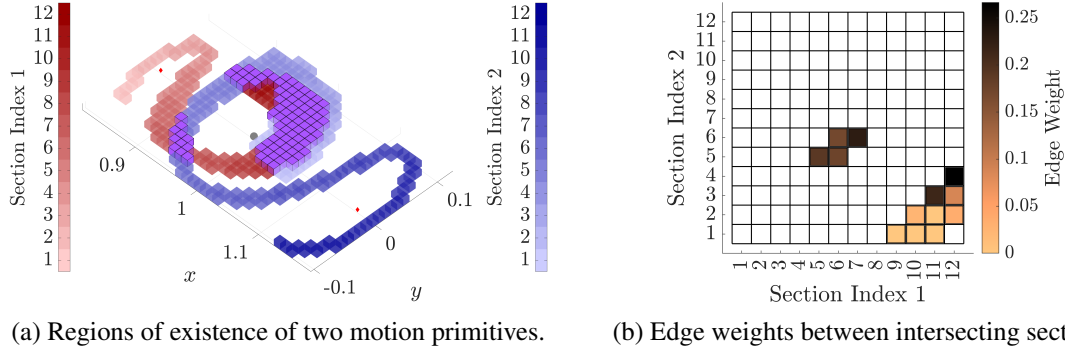


Figure 6: Identifying intersections between the regions of existence of two behavioral motion primitives to estimate their potential for sequential composability.

Directed edges are added between selected nodes associated with intersecting motion primitives that are considered sequentially composable. To identify intersections between each pair of motion primitives in the library, the center of every position voxel from one primitive’s region of existence is compared to the centers of the position voxels from the other primitive’s region of existence. If two voxel centers overlap, those voxels and the corresponding sections in each region are marked as intersecting. For each unique combination of section indices along the regions of existence, bidirectional edges are added between the associated nodes in the graph to reflect that initial guesses could be constructed using these two sections sequentially composed in either order.

To calculate the weight of the edge connecting section a of one primitive and section b of the other primitive, the pairs of intersecting position voxels between sections a and b are identified and a set of n_p velocity pairs is extracted from the associated sets of velocity voxels. Then, the edge weight between the two sections $w_{a,b}$ is defined using the minimum relative velocity discontinuity over all velocity pairs as

$$w_{a,b} = \min_{1 \leq k \leq n_p} \frac{\|\mathbf{v}_i^k - \mathbf{v}_j^k\|}{\|\mathbf{v}_i^k\| + \|\mathbf{v}_j^k\|}. \quad (8)$$

where $\mathbf{v}_i$ is a velocity from a position voxel in section a and $\mathbf{v}_j$ is a velocity from a position voxel in section b that intersects the voxel from section a . Scaling the edge weight by the speeds focuses on analyzing relative changes in the velocity, which is useful when generating sequences of primitives that may pass close to the Moon with higher speeds.¹⁵ Figure 6(b) summarizes the calculated edge weight of each combination of composable sections from the two primitives plotted in Figure 6(a). Two groups of intersecting sections are visible in the grid: a smaller, higher edge weight group and a larger, lower edge weight group. These groupings are consistent with Figure 6(a) where there is a large purple voxel intersection region with well-aligned velocity vectors and a smaller purple voxel intersection region with larger differences in the velocity directions. For inter-primitive edges with a calculated edge weight of zero ($\mathbf{v}_i = \mathbf{v}_j$), the actual edge weight is set to 10^{-14} to facilitate later graph searches prioritizing flow edges over inter-primitive edges where appropriate.

As previously demonstrated by Miceli and Bosanac,¹⁵ additional path or maneuver constraints can be applied when calculating edge weights. In this paper, a maximum velocity angle difference of 30° is applied¹⁵ except for velocity pairs where both velocity norms are below 10 m/s. This constraint avoids populating the graph with edges that would require large direction changes and unnecessarily high impulsive maneuver costs.

Step 4: Generating Motion Primitive Sequences

The motion primitive graph is searched to identify motion primitive sequences that support generating long-term trajectory predictions from an uncertain state estimate. The estimation mapping process provides a list of start nodes, corresponding to sections along primitives that overlap with the uncertainty hyperellipsoid. Sequences of motion primitives that begin from these nodes support generating a summary of reachable orbits, regions, or vantage points. In this proof of concept, a set of candidate destinations are selected as libration point and Moon-centered periodic orbits, the Moon’s surface, and the L_1 and L_2 gateways. A primitive sequence that connects the initial uncertainty hyperellipsoid to any of these destinations is used to generate a continuous trajectory in the next step and, therefore, supply a single long-term trajectory prediction. Furthermore, the collection of primitive sequences supplies geometrically distinct pathways to reachable destinations, potentially supplying a digestible set of long-term trajectory predictions.

The end node is specified before each graph search. If the destination is a periodic orbit group, its associated region of existence is located. Then, one node corresponding to a section along the primitive is selected. The nodes of the periodic orbit group in the graph can be freely traversed in a loop, so the search algorithm will naturally select the optimal node for insertion onto the orbit. If, however, the destination is the Moon’s surface, then a ring of position voxels each with a side length of 1 km is placed at the Moon’s surface to target a close approach to the Moon with an altitude below approximately 500 m. This ring of position voxels is represented as a single node in the graph and marked as the end node. Finally, if the destination is the L_1 or L_2 gateway, then a line of position voxels is placed in the $(\hat{x}, \hat{y})$ plane and parallel to the $\hat{y}$ axis at $x = 0.75$ or $x = 1.23$, respectively. These lines of position voxels span $y = [-0.125, 0.125]$ to target a variety of escape geometries and are each assigned a single node in the graph. The corresponding node is selected as the end node.

This paper uses Yen’s algorithm to generate multiple paths through the graph between selected start and end nodes. These paths correspond to sequences of nodes and edges that are transformed to primitive sequences. Within Yen’s algorithm, Dijkstra’s algorithm is used to compute each candidate path with MATLAB’s *shortestpath*.

For each of the 18 initial primitives that intersect the uncertain state estimate in Figure 4, up to 50 paths are generated to each of the 22 destinations by searching the motion primitive graph. This search process results in 10,212 paths that require approximately two hours to compute. Two of these primitive subsequences are plotted in Figure 7. In each subfigure, the series of sections from each region of existence selected by the graph search are plotted and uniquely colored for each primitive; darker shades indicate increasing section indices. Sections along the first primitive in the sequence, which intersect the observation uncertainty hyperellipsoid centered at the blue star, are colored purple. In Figure 7(a), the primitive subsequence reaches an L_1 Lyapunov orbit. In Figure 7(b), however, the primitive subsequence escapes through the L_2 gateway. Both motion primitive subsequences connect relatively smoothly, increasing the likelihood of producing a good initial guess for a trajectory with impulsive maneuvers.

Step 5: Generating an Initial Guess from a Primitive Sequence

To generate initial guesses, the sequences of primitive sections are processed. This refinement process is implemented in a manner similar to Miceli and Bosanac,¹⁵ through the construction of a low-level graph that uses states along an array of arcs from each region of existence for nodes. A representative subset of the arcs that lie within the region of existence of each primitive in a

sequence is identified by clustering in the position-based feature vector space using the k -medoids algorithm, accessed in MATLAB.²⁵ This clustering step samples the set of arcs associated with each primitive to reduce the low-level graph size and computational costs. Then, the sampled states along each selected arc define the nodes in the low-level graph. States that lie within 10 km of the Moon’s surface are not incorporated in the graph to support the computation of feasible initial guesses.

Edges are added between nodes associated with states along the same arc and between composable states from sequential primitives. Zero-weight, directed edges are added between nodes defined using sequential states sampled from the same arc, ensuring that arcs can be traversed without penalty. Similar zero-weight edges are added from the last node to the first node along arcs following periodic orbits to encode periodicity. Directed, weighted edges are then added between nodes on distinct primitives that are sequentially composable. These edge weights W between the nodes are calculated by weighting position and velocity discontinuities as

$$W = w_p |\Delta \mathbf{r}| + w_v (1 - \cos \theta), \quad (9)$$

where $\Delta \mathbf{r}$ is the position difference between two corresponding states and θ is the angle between their velocity vectors. Scalar coefficients $w_p = 10$ and $w_v = 1$ balance the relative importance of connections with close proximity or well-aligned velocity directions, respectively. Edges are added between trajectories of each primitive set according to the computed path through the high-level motion primitive graph. As a result, at each node of the graph the only choices are to continue along the current arc or move to a state along an arc associated with the next primitive in the sequence.

The modified A* search algorithm, developed by Miceli and Bosanac,¹⁵ is used to search this graph for a sequence of nodes and edges that minimizes the cumulative edge weight when traversing from a state nearby the observation to a selected primitive in the destination set. One modification requires that the path consist of at least two nodes along any arc, i.e., every arc must be traversed for a nonzero duration. A list of potential start nodes are selected from the arcs of the primitive set that intersect the uncertainty hyperellipsoid. From this set, one potential start node per arc is selected as the state closest to the mean estimate. The list of multiple start nodes is incorporated by adding them all to the priority queue when initializing the modified A* algorithm. A list of potential end nodes is composed of the final nodes along each representative arc corresponding to the selected primitive

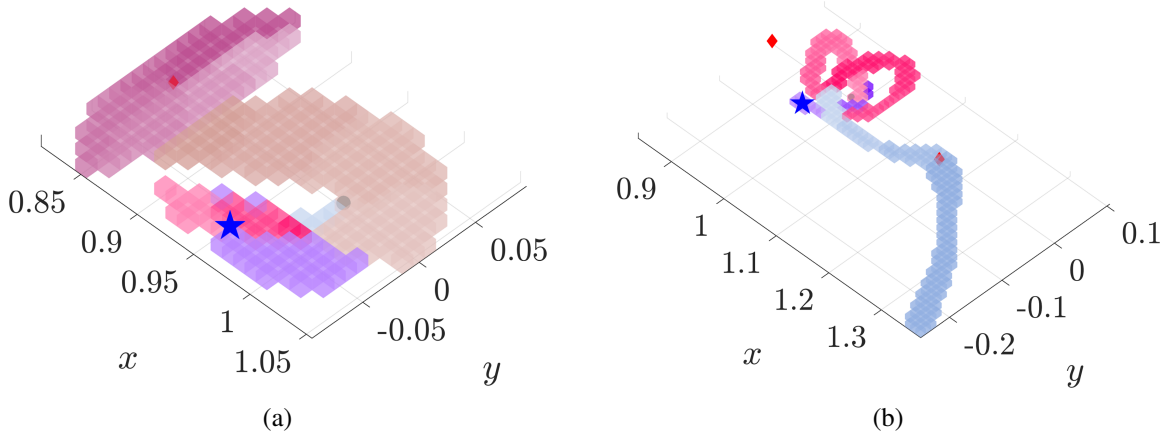


Figure 7: Two primitive subsequences from the observation to (a) an L_1 Lyapunov orbit primitive and (b) L_2 gateway escape.

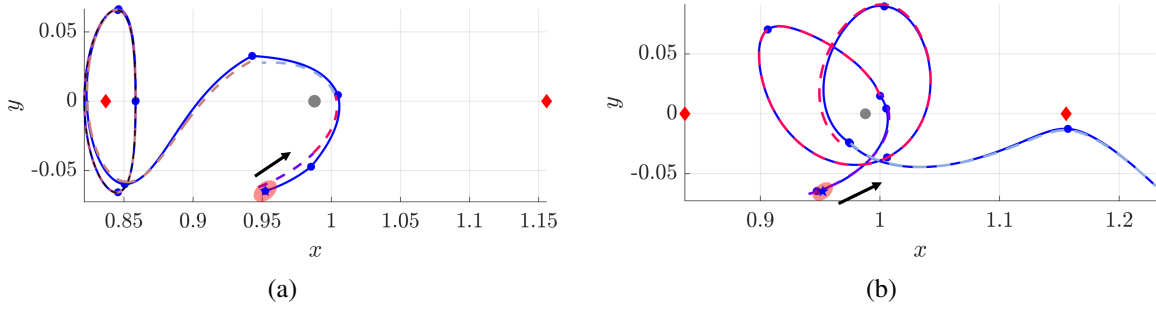


Figure 8: Two discontinuous initial guesses in dashed curves and the corrected trajectories in solid curves: (a) from the observation to an L_1 Lyapunov orbit and (b) from the observation to an escape through the L_2 gateway.

from the destination set. Accordingly, an additional modification to A* is the allowance of multiple end nodes by 1) allowing termination upon reaching any end node and 2) setting an A* heuristic $h(n)$ that considers all possible end nodes. The selected A* heuristic is the minimum shortest path distance from the current node n to any end node. This cost is calculated without considering the two node sub-path constraint and will always be less than or equal to the true cost when accounting for the constraint to satisfy the admissible heuristic criteria for A*.

Of the 10,212 high-level paths generated, 9,100 of them are successfully refined to produce initial guesses in under two hours. A failure to refine indicates that the A* algorithm failed to find a viable path through the low-level graph with at least two states per primitive that satisfies the 10 km minimum Moon altitude constraint. Two refined initial guesses computed from the primitive subsequences in Figure 7 are plotted in Figure 8 with dashed curves. Each unique color in the initial guess corresponds to a different segment of a primitive. These initial guesses exhibit a wide variety of geometries with small position discontinuities and velocity direction changes.

Step 5: Correcting Initial Guesses Using Multiple Shooting

Each discontinuous initial guess is corrected via multiple-shooting to produce a continuous trajectory with impulsive maneuvers. Initial guesses for impulsive maneuvers are placed at maxima in the curvature and at the boundaries of arcs from distinct primitives.¹⁵ At these locations, only position constraints are enforced. Two samples are evenly distributed in the arclength between subsequent impulsive maneuvers, and full state continuity is enforced between these smaller arcs. In addition, the initial position is constrained to equal the observation mean, $\bar{x}$, and the final position is constrained to match the final position of the initial guess, $\mathbf{r}_{f,IG}$. Further constraints are added to ensure that periapses maintain an altitude of 10 km above the Moon's surface. The vector of additional path constraints, formed by adding two constraints and one slack variable β_{p_j} for the j th of m identified periapsis events, is

$$\mathbf{F}_{p,j} = \left[(\mathbf{r}_{p,j} - \mathbf{r}_{moon}) \cdot \mathbf{v}, \quad \|\mathbf{r}_{p,j} - \mathbf{r}_{moon}\| - r_{moon,rad} - \beta_{p_j}^2 \right]^T. \quad (10)$$

This implementation assumes that for a close initial guess, a periapsis event will not become an apoapsis event during corrections and, therefore, the sign of the time derivative of the first constraint is not incorporated into $\mathbf{F}_{p,j}$. Additional constraints enforce a non-negative duration of each arc

using slack variables β_{t_i} for arcs $i = 1, 2, \dots, n$. The additional constraint vectors applied to the entire initial guess are defined as

$$\mathbf{F}_{BCs} = \begin{bmatrix} \mathbf{x}_{1,0} - \bar{\mathbf{x}} \\ \mathbf{r}_{n,f} - \mathbf{r}_{f,IG} \end{bmatrix}, \quad \mathbf{F}_p = \begin{bmatrix} \mathbf{F}_{p,1} \\ \vdots \\ \mathbf{F}_{p,m} \end{bmatrix}, \quad \text{and} \quad \mathbf{F}_t = \begin{bmatrix} \Delta T_1 - \beta_{t_1}^2 \\ \vdots \\ \Delta T_n - \beta_{t_n}^2 \end{bmatrix}. \quad (11)$$

Together, all the constraints are used to define the full constraint vector as $\mathbf{F} = [\mathbf{F}_{BCs}^T \mathbf{F}_0^T \mathbf{F}_p^T \mathbf{F}_t^T]^T$.

Each initial guess, derived from a primitive sequence, is corrected. During the corrections process, the constraint vector must reach a norm of less than or equal to $\tau = 10^{-10}$ within 100 iterations. Of the 9,100 initial guesses generated, 7,755 of them are successfully corrected to produce continuous trajectories in about twenty minutes. Two corrected trajectories are plotted with solid curves in Figure 8, overlaid with their corresponding initial guesses and marked with circles to indicate impulsive maneuver locations. The corrected trajectories retain the geometry of their initial guesses, but are now continuous at each arc intersection in the configuration space.

Step 6: Generating a Digestible Trajectory Prediction Summary

Generating a digestible summary of possible trajectories from an observation requires a reduction in the total number of trajectories. To achieve this goal, their similarity is assessed using dynamic time warping (DTW).^{38,39} DTW is a distance measure that is typically used in time series analysis for comparing signals of varying speed, pairing similar samples from two sets to minimize the sum of Euclidean distances between the pairs.³⁹ A sample from one set may be paired with multiple, sequential samples from the other set if necessary to produce the minimum sum of distances.

First, the corrected trajectories are grouped by destination and sorted by their total maneuver magnitude. The trajectory with the minimum total maneuver magnitude from each group initializes a list of sufficiently unique summarizing trajectories. Each additional trajectory is compared to the trajectories in the list through two separate applications of dynamic time warping, in position and velocity, to determine if it is sufficiently different to be added to the final list. If the position and velocity similarity metrics, calculated via DTW, are both above a manually selected similarity threshold, the trajectory is considered sufficiently unique and added to the list for its respective group. The similarity thresholds are tuned for the individual destinations and range from 1-2 and 2-10 nondimensional units for position and velocity, respectively.

Step 7: Natural Parameter Continuation for Maneuver Requirement Reduction

Natural parameter continuation is used to reduce maneuver requirements of each summarizing trajectory and produce families of feasible paths; this step is modeled after the process used by Spear and Bosanac.⁴⁰ To retain geometric similarity to the initial guess and enforce necessary constraints, the multiple shooting algorithm is modified to incorporate a multi-objective cost function¹²

$$J(\mathbf{V}) = w_{geo}(\mathbf{V}_{pos} - \mathbf{V}_{ref,pos})^T(\mathbf{V}_{pos} - \mathbf{V}_{ref,pos}) + w_{man} \sum_{i=1}^{n_{man}} \Delta v_i^2, \quad (12)$$

where w_{geo} weights geometric similarity to the initial guess, w_{man} weights the sum of squared maneuver magnitudes, n_{man} is the number of maneuvers, $\mathbf{V}_{pos}$ contains the position elements of the current free variable vector $\mathbf{V}$, and $\mathbf{V}_{ref}$ contains the position elements of the free variable vector for

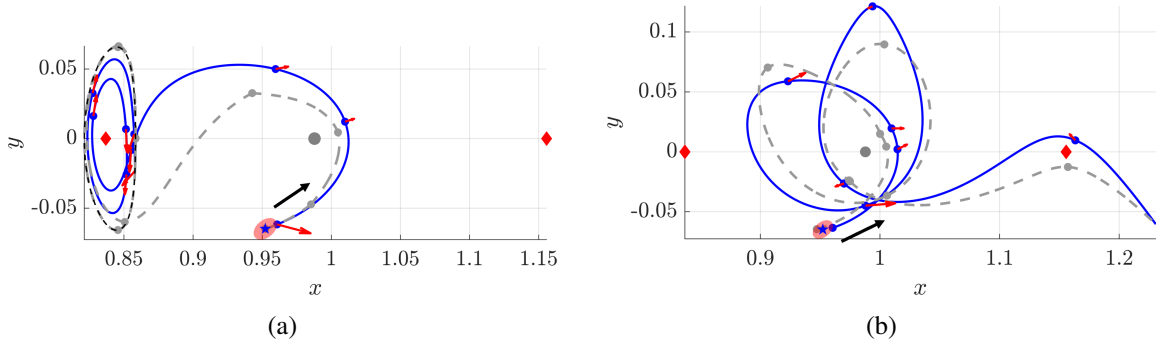


Figure 9: Two sets of corrected and continued trajectories: (a) from the observation to an L_1 Lyapunov orbit and (b) from the observation to an escape through the L_2 gateway.

the original corrected trajectory, which is labeled as the reference trajectory. The added constraint takes the form $\mathbf{F}_J = J(\mathbf{V}) - J_{des}$, where J_{des} is a desired value of J . Additionally, $\mathbf{F}'_{BCs}$ is defined as follows to enforce that $\mathbf{x}_{1,0}$ can lie anywhere within the uncertainty hyperellipsoid using the Mahalanobis distance and a slack variable β_e :

$$\mathbf{F}'_{BCs} = \left[(\mathbf{x}_{1,0} - \bar{\mathbf{x}})^T \Sigma^{-1} (\mathbf{x}_{1,0} - \bar{\mathbf{x}}) - \sigma^2 + \beta_e^2, (\mathbf{r}_{n,f} - \mathbf{r}_{f,IG})^T \right]^T \quad (13)$$

The full constraint vector during continuation is defined as $\mathbf{F}_{cont} = [\mathbf{F}'_{BCs}^T \mathbf{F}_0^T \mathbf{F}_p^T \mathbf{F}_t^T \mathbf{F}_J^T]^T$.

A two-step continuation method is used, as presented by Spear and Bosanac.⁴⁰ In an outer loop, the weights are varied from $[w_{geo}, w_{man}] = [0.95, 0.05]$ to $[0, 1]$. This continuation guides the corrections algorithm to converge towards a “minimum energy” solution. In an inner loop, the value of J_{des} is adaptively lowered by ΔJ_{des} for the current combination of w_{geo} and w_{man} and the trajectory is corrected. If a solution is not generated within 30 iterations, corrections are terminated. Then, J_{des} is reset to its previous value, and the step size of J_{des} is halved. The inner loop terminates once the step size drops below $\frac{1}{10}$ th of the initial step size $\Delta J_{des,0} = 10^{(\lfloor \log_{10}(J_0) \rfloor - 1)}$, where J_0 is the initial value of J for a specified w_{geo} and w_{man} . This process reduces maneuver requirements by a similar order of magnitude as optimization via MATLAB’s *fmincon* for these trajectories, while offering quicker computation times.

Examples of trajectories with reduced maneuver requirements are shown in Figure 9. In this figure, the corrected trajectories from Figure 8 are showed in dashed gray, the continued trajectories with $w_{geo} = 0$ and $w_{man} = 1$, are shown in blue, and red arrows indicate the maneuver directions and relative magnitudes. The blue trajectory in Figure 9(a) requires a total Δv of 99.0 m/s and a time of flight of 28.8 days, whereas the blue trajectory in Figure 9(b) requires a total Δv of 151.8 m/s and a time of flight of 18.6 days. Although these solutions are not truly optimal in an energy or Δv sense, they are sufficient for the generation of preliminary trajectory predictions.

RESULTS AND DISCUSSION

Trajectory predictions from the uncertain state estimate in Figure 4 to a variety of destinations in the lunar vicinity are generated using the process detailed in this paper. Table 1 summarizes the ranges of total maneuver costs, time of flight (TOF), and final Jacobi constants $C_{J,f}$ for the predicted possible trajectories to each destination, i.e., a segment of a periodic orbit family or specific

termination condition; in this section, each destination is uniquely labeled using a lowercase letter. The range of destination orbit periods P are also included, where applicable. Across the diverse set of generated trajectories, total maneuver magnitudes vary from approximately 2.1 m/s to 504.4 m/s, flight durations span from about 0.7 days to 80.7 days, and final Jacobi constants range between 2.29 and 3.22. For reference, the Jacobi constant of the observation estimate mean $\bar{x}$ is 3.19.

Table 1: Ranges of Δv (m/s), time of flight (days), final Jacobi constant, and period of destination periodic orbit, where applicable, for each destination (dest.) in Figures 10 and 11.

Dest.	L_1 Lyapunov				Distant Retrograde	
	a	b	c	d	e	f
Δv	[2.1, 206.0]	[95.7, 468.8]	[257.2, 359.3]	[258.6, 504.4]	[161.9, 468.7]	[253.1, 592.9]
TOF	[15.7, 58.8]	[17.9, 39.8]	[34.5, 40.5]	[36.5, 45.3]	[14.1, 37.2]	[38.5, 61.0]
$C_{J,f}$	[3.16, 3.19]	[2.96, 3.15]	[2.91, 2.95]	[2.72, 2.91]	[2.97, 3.40]	[2.29, 2.70]
P	[11.7, 12.1]	[12.3, 23.6]	[24.1, 28.2]	[28.3, 32.4]	[0.9, 9.2]	[26.0, 27.2]

Dest.	L_2 Lyapunov			Distant Prograde		
	g	h	i	j	k	l
Δv	[24.4, 317.9]	[114.6, 169.2]	[93.9, 186.9]	[98.1, 342.0]	[93.6, 142.2]	[107.5, 196.7]
TOF	[27.7, 62.0]	[35.2, 57.5]	[25.0, 43.4]	[4.8, 22.9]	[12.8, 32.6]	[20.6, 61.5]
$C_{J,f}$	[3.15, 3.17]	[3.03, 3.14]	[2.90, 2.96]	[3.16, 3.18]	[3.10, 3.15]	[3.09, 3.10]
P	[14.7, 14.8]	[14.9, 17.8]	[24.4, 32.8]	[5.9, 7.9]	[8.7, 13.7]	[14.0, 15.1]

Dest.	Distant Prograde			Low Prograde		
	m	n	o	p	q	r
Δv	[77.3, 113.6]	[58.0, 136.4]	[62.0, 211.0]	[154.7, 329.9]	[4.6, 49.0]	[4.9, 73.2]
TOF	[18.2, 53.2]	[29.2, 79.3]	[36.3, 80.7]	[2.8, 13.9]	[6.9, 29.6]	[10.7, 30.6]
$C_{J,f}$	[3.00, 3.08]	[2.99, 3.00]	[2.94, 2.99]	[3.19, 3.45]	[3.18, 3.19]	[3.18, 3.18]
P	[16.5, 26.7]	[27.5, 28.6]	[28.9, 40.9]	[1.1, 5.1]	[6.0, 10.0]	[10.9, 11.0]

Dest.	Low Prograde	L_1 Gateway	L_2 Gateway	Moon Impact
	s	t	u	v
Δv	[11.5, 81.1]	[3.3, 118.2]	[18.5, 354.1]	[120.6, 324.5]
TOF	[10.8, 32.9]	[5.2, 12.9]	[16.3, 31.1]	[0.7, 17.3]
$C_{J,f}$	[3.18, 3.18]	[3.16, 3.19]	[3.04, 3.16]	[3.17, 3.22]
P	[11.8, 11.9]	—	—	—

The predicted trajectories are plotted in Figures 10 and 11, respectively. Each trajectory is plotted in a distinct color, final periodic orbits are drawn with black dashed lines, circle markers indicate impulsive maneuver locations, and the libration points are depicted with red diamonds. For particularly extensive paths, adjacent zoomed figures provide additional detail.

The generated summary includes long-term trajectory predictions with a wide variety of geometries, including multiple lunar revolutions, direct gateway escapes, and Moon impacts. For comparison, traditional methods such as Poincaré mapping⁴¹ and Monte Carlo⁴ sampling, while effective for many situations, can be time-consuming and produce an overwhelming volume of data, ren-

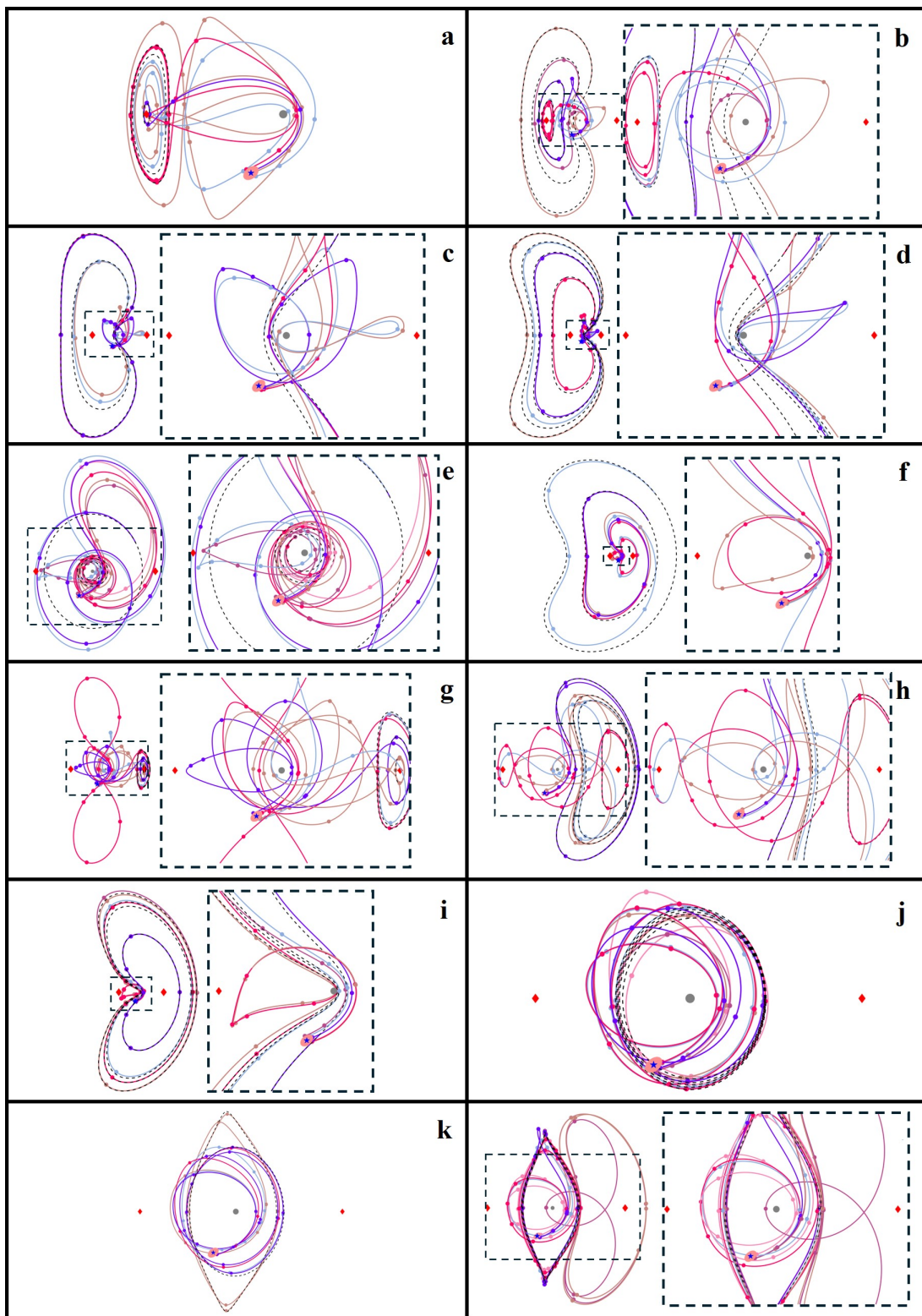


Figure 10: Trajectory predictions for destinations a through l

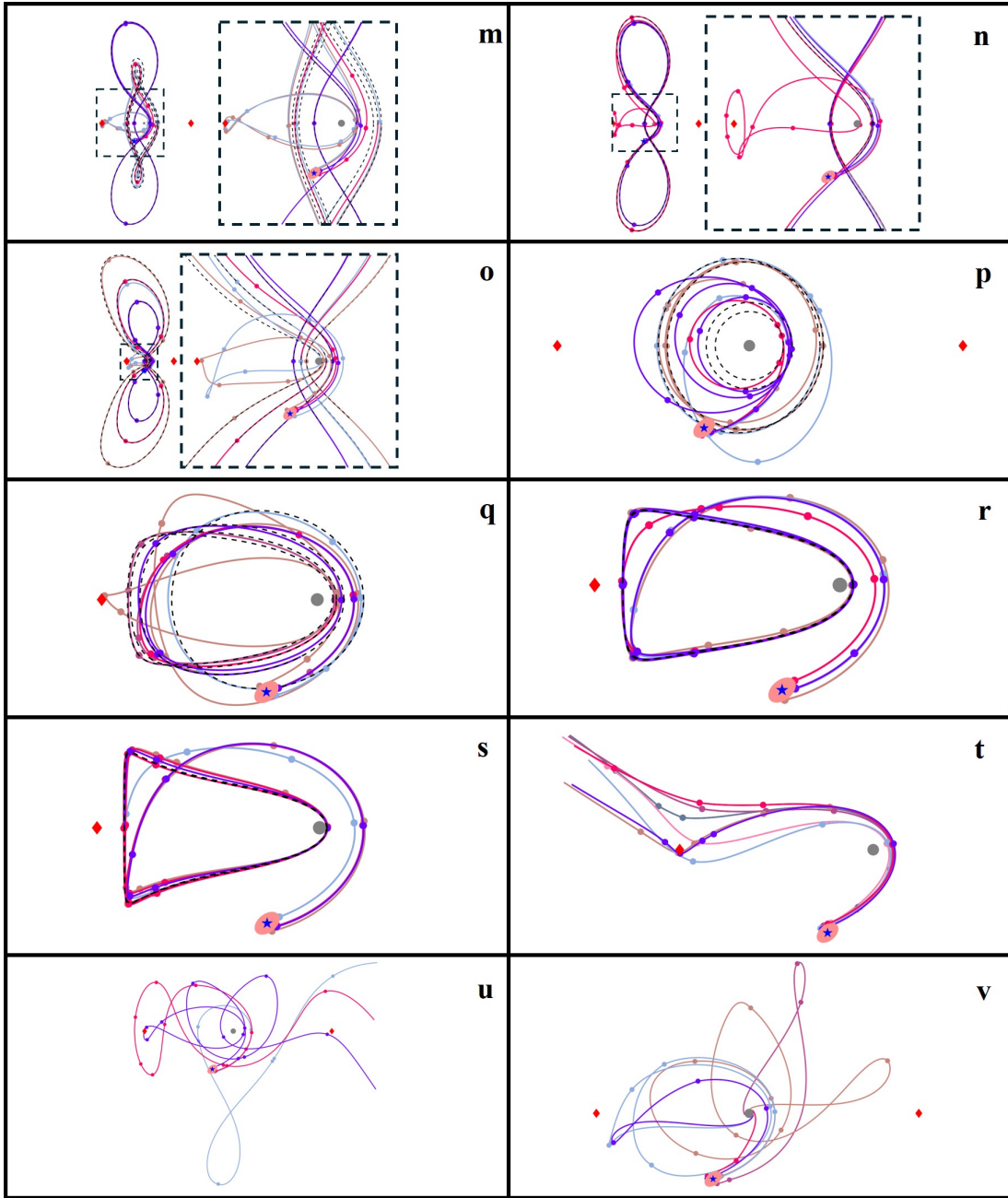


Figure 11: Trajectory predictions for destinations m through v

dering them impractical for rapid decision-making. Poincaré maps require extensive computation and manipulation to identify reachable sets, whereas Monte Carlo approaches lack a natural mechanism to incorporate intentional maneuvers or destination constraints. In contrast, the behavioral motion primitive framework delivers concise, destination-focused summaries by deliberately generating geometrically distinct transfers. The trajectory predictions that reach periodic orbits delineate accessible vantage points suitable for long-duration missions. However, the L_1 and L_2 gateway escape prediction sets highlight available departure options or warn of inadvertent exits from the

lunar region. Similarly, the lunar impact trajectory set could inform risk assessments or planned impact strategies, depending on mission objectives. Due to the current absence of a maximum maneuver constraint in the high-level graph search, some high maneuver cost trajectories are returned. These paths may not be feasible for every spacecraft, but still contribute important information to the prediction summary.

CONCLUSION

This paper presents a new approach to long-term spacecraft trajectory prediction in cislunar space by leveraging a library of motion primitives. Each primitive’s region of existence is approximated via discrete voxels with encoded velocity information, enabling rapid mapping of uncertain state estimates onto a set of short-term motion predictions. By constructing and searching a motion primitive graph, sequences of composable primitives are generated from the initial state estimate to various vantage points. Diverse trajectory prediction sets are generated in the CR3BP for destinations including 19 periodic orbit primitives, L_1 and L_2 gateway escapes, and lunar impacts. Future work will extend this framework to spatial and continuous thrust primitives.

ACKNOWLEDGMENT

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-23-1-0235. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the United States Air Force.

REFERENCES

- [1] P. R. Escobal, *Methods of Orbit Determination*, pp. 239–292. Malabar, FL: Krieger, 1965.
- [2] R. H. Gooding, “A New Procedure for Orbit Determination Based on Three Lines of Sight (Angles Only),” Tech. Rep. TR-93004, Defense Research Agency, Farnborough, Hampshire, England, 1993.
- [3] B. D. Tapley, B. E. Schutz, and G. H. Born, “Chapter 4 - Fundamentals of Orbit Determination,” *Statistical Orbit Determination*, pp. 159–284, Burlington: Academic Press, 2004.
- [4] Y. z. Luo and Z. Yang, “A Review of Uncertainty Propagation in Orbital Mechanics,” *Progress in Aerospace Sciences*, Vol. 89, 2017, pp. 23–39.
- [5] A. Wolek and C. A. Woolsey, *Model-Based Path Planning*. Springer International Publishing, 2017.
- [6] O. Dermay, A. Paraschos, M. Ewerton, J. Peters, F. Chappillet, and S. Ivaldi, “Prediction of Intention during Interaction with iCub with Probabilistic Movement Primitives,” *Frontiers in Robotics and AI*, Vol. 4, 2017.
- [7] D. Clever, M. Harant, H. Koch, K. Mombaur, and D. Endres, “A Novel Approach for the Generation of Complex Humanoid Walking Sequences Based on a Combination of Optimal Control and Learning of Movement Primitives,” *Robotics and Autonomous Systems*, Vol. 83, 2016, pp. 287–298.
- [8] S. Ferguson, B. Luders, R. C. Grande, and J. P. How, “Real-Time Predictive Modeling and Robust Avoidance of Pedestrians with Uncertain, Changing Intentions,” *CoRR*, 2014.
- [9] A. Majumdar and R. Tedrake, “Funnel Libraries for Real-Time Robust Feedback Motion Planning,” *CoRR*, 2016.
- [10] G. Habibi, N. Jaipuria, and J. P. How, “SILA: An Incremental Learning Approach for Pedestrian Trajectory Prediction,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 4411–4421.
- [11] H. Edelhoff, J. Signer, and N. Balkenhol, “Path Segmentation for Beginners: an Overview of Current Methods for Detecting Changes in Animal Movement Patterns,” *Movement Ecology*, Vol. 4, No. 1, 2016.
- [12] T. R. Smith and N. Bosanac, “Motion Primitive Approach to Spacecraft Trajectory Design in a Multi-body System,” *The Journal of the Astronautical Sciences*, Vol. 70, No. 34, 2023.
- [13] N. Bosanac, “Clustering Natural Trajectories in the Earth-Moon Circular Restricted Three-Body Problem,” 2025, Accepted to The Journal of the Astronautical Sciences.

- [14] C. Gillespie, G. E. Miceli, and N. Bosanac, “Summarizing Natural and Controlled Motion in Cislunar Space With Behavioral Motion Primitives,” *AAS/AIAA Space Flight Mechanics Meeting*, 2025.
- [15] G. E. Miceli and N. Bosanac, “Generating Planar Trajectories for Neptunian System Exploration Using Motion Primitives,” 2025, Under Review.
- [16] G. E. Miceli and N. Bosanac, “Designing Neptunian System Tours via a Motion Primitive Approach,” *AAS/AIAA Space Flight Mechanics Meeting*, 2025.
- [17] V. Szebehely, *Theory of Orbits: The Restricted Problem of Three Bodies*. London: Academic Press, 1967.
- [18] R. S. Park, W. M. Folkner, J. G. Williams, and D. H. Boggs, “The JPL Planetary and Lunar Ephemerides DE440 and DE441,” *The Astronomical Journal*, Vol. 161, 2021, p. 105.
- [19] K. Wardle, *Differential Geometry*. Mineola, NY: Dover Publications, Inc., 2008.
- [20] N. Patrikalakis, T. Maekawa, and W. Cho, *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer, Berlin, Heidelberg, 2009. E-book.
- [21] N. Bosanac, “Curvature Extrema Along Trajectories in the Circular Restricted Three-Body Problem,” *AAS/AIAA Astrodynamics Specialists Conference*, 2024.
- [22] J. Han, M. Kamber, and J. Pei, *Data Mining (Third Edition)*. The Morgan Kaufmann Series in Data Management Systems, Boston: Morgan Kaufmann, third edition ed., 2012.
- [23] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD’96, AAAI Press, 1996, p. 226–231.
- [24] R. J. G. B. Campello, D. Moulavi, and J. Sander, “Density-Based Clustering Based on Hierarchical Density Estimates,” *Advances in Knowledge Discovery and Data Mining* (J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, eds.), Berlin, Heidelberg, Springer Berlin Heidelberg, 2013, pp. 160–172.
- [25] T. M. Inc., “MATLAB version: 24.1.0 (R2024a),” 2024.
- [26] C. Malzer and M. Baum, “A Hybrid Approach To Hierarchical Density-based Cluster Selection,” *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 2019, pp. 223–228.
- [27] L. McInnes, J. Healy, and S. Astels, “hdbscan: Hierarchical Density Based Clustering,” *The Journal of Open Source Software*, Vol. 2, mar 2017.
- [28] K. HOWELL and H. PERNICKA, “Numerical Determination of Lissajous Trajectories in the Restricted Three-body Problem,” *Celestial Mechanics*, Vol. 41, 1987, pp. 107–124.
- [29] A. Kaufman, D. Cohen, and R. Yagel, “Volume Graphics,” *Computer*, Vol. 26, No. 7, 1993, pp. 51–64.
- [30] K. C. Howell, M. Beckman, C. Patterson, and D. Folta, “Representations of Invariant Manifolds for Applications in Three-Body Systems,” *The Journal of the Astronautical Sciences*, Vol. 54, Mar 2006, pp. 69–93.
- [31] W. Dijkstra, E. “A Note on Two Problems in Connexion with Graphs,” *Numerische Mathematik*, Vol. 1, Dec 1959, pp. 269–271.
- [32] E. Hart, P. J. Nilsson, N. and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 2, 1968, pp. 100–107.
- [33] J. Y. Yen, “An Algorithm for Finding Shortest Routes from All Source Nodes to a Given Destination in General Networks,” *Quarterly of Applied Mathematics*, Vol. 27, 1970, pp. 526–530.
- [34] J. S. Parker and R. L. Anderson, *Low-energy Lunar Trajectory Design*. Wiley, 2014.
- [35] N. Bosanac, “Data-Driven Summary of Motion in an Ephemeris Model of Cislunar Space,” *AAS/AIAA Space Flight Mechanics Meeting*, Lihue, HI, 2025.
- [36] Y. Zheng and X. Zhou, *Computing with Spatial Trajectories*. Springer Publishing Company, Incorporated, 1st ed., 2011.
- [37] P. C. Mahalanobis, “On the Generalized Distance in Statistics,” *Proceedings of the National Institute of Sciences of India*, Vol. 2, 1936, pp. 49–55.
- [38] K. L. Bruchko and N. Bosanac, “Rapid Trajectory Design in Multi-Body Systems Using Sampling-Based Kinodynamic Planning,” *The Journal of the Astronautical Sciences*, Vol. 72, July 2025.
- [39] M. Müller, “Dynamic Time Warping,” *Information Retrieval for Music and Motion*, pp. 69–84, Springer, Berlin, Heidelberg, 2007.
- [40] R. L. Spear and N. Bosanac, “Tree-Based Approach to Spacecraft Trajectory Design in Multi-Body Systems,” 2025, Under Review.
- [41] L. Perko, *Differential Equations and Dynamical Systems*, Vol. 7 of *Graduate Texts in Mathematics*. New York, NY: Springer-Verlag, 2nd ed., 1996.