

AUTOMATICALLY GENERATING TRAJECTORIES FOR A DISTRIBUTED SPACECRAFT MISSION IN A MULTI-BODY SYSTEM

Cole Gillespie* and Natasha Bosanac[†]

This paper focuses on automatically generating conflict-free paths for a distributed spacecraft mission in a multi-body system by using motion primitives in a conflict-based search. A library of motion primitives summarizes natural motion in a selected region of the Earth-Moon circular restricted three-body problem. These primitives support rapid constraint evaluation, are geometrically distinct, and can be automatically sequenced to form initial guesses for each spacecraft path. Conflict-free initial guesses that minimize the total maneuver requirements are efficiently generated using a conflict-based search. These initial guesses are then corrected to produce continuous trajectories for multiple spacecraft.

INTRODUCTION

A distributed spacecraft mission (DSM) is composed of a set of multiple spacecraft that cooperatively achieve a common mission goal.¹ DSMs may be useful for scientific, exploration, and infrastructure objectives in multi-body regimes throughout the solar system. For example, a network or constellation of communication and navigation satellites could support a growing number and cadence of cislunar operations.² Alternatively, DSMs could support remote sensing missions of planetary bodies. In these cases, trajectories must be designed for multiple spacecraft to satisfy their individual constraints in a sensitive dynamical environment while also achieving the goals of the DSM and, potentially, with similar maneuver requirements. This trajectory design task may need to be repeated throughout the mission design process and in time-critical reconfiguration scenarios. Automating a rapid and efficient approach to solving this trajectory design problem can support reducing the workload of human decision-makers as well as autonomous operations.

Traditional DSM trajectory design strategies such as Walker³ or Streets-of-Coverage^{4,5} constellations and more geometric methods⁶ have been used to meet coverage requirements in two-body systems. Reconfiguration strategies for such constellations have also been studied.⁷ However, these designs are not easily adaptable to multi-body systems. More recently, the HelioSwarm mission used lunar resonant orbits to achieve recurring geometries of its satellites through its trajectory design.⁸ Creating a general framework for designing trajectories for multiple, cooperative spacecraft could support expanding the array of DSMs throughout the solar system.

Multi-agent path-planning has been studied in the robotics community, with applications including self-driving cars, robot swarms, or warehouse robots. In these scenarios, multiple robots must

*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.

traverse an environment to meet a goal while avoiding collisions and obstacles.⁹ There are multiple approaches to solving this path-planning problem, with various levels of computational complexity and likelihood of recovering a feasible solution. Coupled path-planning approaches consider designing trajectories for all the robots simultaneously in a higher-dimensional solution space at the expense of higher computational time and burden.¹⁰ Alternatively, decoupled path-planning approaches generate a set possible paths for each robot and then identify combinations of feasible paths from this smaller set of options.⁹ Depending on the approach, the computational complexity may be reduced at the expense of producing suboptimal paths.¹¹

The capability to automatically generate geometrically distinct paths for each member of a multi-agent system supports definition of a structure to efficiently organize configuration guesses, evaluate conflicts, and prioritize replanning, known as a conflict tree. One existing approach to path-planning that may provide these benefits involves the use of motion primitives, which are defined as discrete “building block” representations of possible motions, actions, or tasks.¹² Sequences of these motion primitive support rapidly generating complex motion plans with distinct characteristics.¹² Motion primitives have been applied to a wide variety of problems such as predicting pedestrian motion or path-planning for a small helicopter.^{13–15}

In recent years, the concept of a motion primitive has been applied to spacecraft trajectory design in multi-body gravitational systems. Smith and Bosanac defined motion primitives to summarize geometrically distinct arcs that exist in similar regions of the phase space, extracting them from periodic orbit families and hyperbolic invariant manifolds via clustering.¹⁶ Gillespie, Miceli, and Bosanac¹⁷ have recently improved upon this procedure by using differential geometry and updated clustering methods to generate higher quality and more diverse motion primitives summarizing natural and controlled motions in a multi-body system. Bodin, Bosanac, and Gillespie have also introduced voxelized regions of existence that summarize the regions of the phase space spanned by geometrically similar arcs near the motion primitive.¹⁸ Smith and Bosanac also implemented trajectory design as a motion primitive sequencing problem by formulating a graph representation, searching the graph, and correcting the resulting initial guesses to produce continuous maneuver-enabled trajectories.¹⁹ More recently, Miceli and Bosanac have substantially improved this process to better reflect the potential for sequentially composing two primitives, formulating a more accurate motion primitive graph, searching the graph to produce geometrically distinct solutions, and generating higher quality initial guesses.²⁰ Minor updates that reduce the computational burden have been presented by Bosanac²¹ as well as Bodin, Bosanac, and Gillespie.¹⁸

This paper focuses on automatically generating a set of conflict-free paths for a distributed spacecraft mission in a multi-body system by using motion primitives in a conflict-based search approach. Leveraging an existing library of natural motion primitives in the Earth-Moon circular restricted three-body problem (CR3BP) along with their voxelized regions of existence,¹⁷ these primitives are used to independently generate a set of geometrically distinct possible primitive sequences for each spacecraft. We then outline the computation of conflicts from the voxel representation of each region of existence along a primitive sequence. Then, this paper presents a formulation of the conflict-based search for DSM trajectory design. Given a set of predefined starting locations and goal locations for each satellite considered in the DSM, we search the conflict tree to construct primitive sequences and associated initial guesses for conflict-free paths. Finally, we correct the set of initial guesses to produce continuous trajectories for each satellite in the DSM.

BACKGROUND

Circular Restricted Three-Body Problem

The circular restricted three-body problem (CR3BP) is a simplified representation of the dynamics within a system of two primary bodies (P_1 and P_2) with a third, much less massive, body orbiting around them.²² Further assumptions of the CR3BP include that the primaries travel on circular orbits around their mutual barycenter with gravitational fields consistent with constant point masses.²² Systems in the CR3BP are characterized by the mass ratio, μ , of the mass of P_2 to the total system mass.²² In the Earth-Moon system, $\mu \approx 0.012151$.

The CR3BP is typically formulated in a frame that rotates with P_1 and P_2 and with physical quantities represented in nondimensional units. The rotating frame is defined with $+\hat{x}$ pointing from P_1 to P_2 , $+\hat{z}$ aligned with the orbital angular momentum vector of the primaries, and $+\hat{y}$ completing the right-handed orthogonal set.²² The origin of this frame is located at the $P_1 - P_2$ barycenter.²² Length values in the Earth-Moon CR3BP are nondimensionalized by the average distance between the Earth and the Moon,²² equal to $l^* = 384,400$ km. Time values are nondimensionalized to produce a mean motion for the orbits of the Earth and Moon that is equal to unity,²² using $t^* \approx 3.751903 \times 10^5$ s. Mass values are nondimensionalized by the total system mass of the Earth and the Moon, $m^* \approx 6.046804 \times 10^{24}$ kg.²³

Equations of motion are written nondimensionally in the rotating frame. Consider the spacecraft state represented by the vector $\mathbf{X} = [\mathbf{r}^T, \mathbf{v}^T]^T = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^T$. The pseudo-potential function U^* is defined as

$$U^* = \frac{1}{2}(x^2 + y^2) + \frac{1 - \mu}{r_1} + \frac{\mu}{r_2} \quad (1)$$

where $r_1 = \sqrt{(x + \mu)^2 + y^2 + z^2}$ and $r_2 = \sqrt{(x - 1 + \mu)^2 + y^2 + z^2}$. Using these definitions, the equations of motion for the spacecraft are expressed as²²

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

The Jacobi constant is defined as²²

$$C_J = 2U^* - \dot{x}^2 - \dot{y}^2 - \dot{z}^2 \quad (3)$$

and is an integral of motion.

The solution space in the CR3BP includes fundamental solutions that have been studied using dynamical systems theory. Lagrange points are equilibrium points in the system numbered L_1 to L_5 .²² Spacecraft located at these points will not depart from them in the rotating frame if their velocity is zero. Periodic orbits are trajectories within the CR3BP that exactly repeat after a minimal time interval, P ,^{22,24} and exist in continuous families. Unstable periodic orbits possess eigenvalues of the monodromy matrix (the state transition matrix propagated for one period) that do not lie on the unit circle.²⁵ These unstable periodic orbits produce stable and unstable manifold pairs that asymptotically approach or depart the periodic orbit.²⁶

Differential Geometry

Geometric investigation of curved paths provides valuable insight into the characterization of trajectories. Physical properties such as arclength and curvature are used in this paper to describe a

trajectory. The arclength is defined as

$$s_{pos} = \int_{t_0}^{t_f} \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} dt \quad (4)$$

and describes the distance traversed along a curve over the time interval $t \in [t_0, t_f]$.²⁷ This quantity provides a parameterization of the curve that is not strongly affected by rapid changes in speed in dynamically sensitive regions of the system. In this paper, an additional scalar quantity is defined over the time interval $t \in [t_0, t_f]$ as

$$s_{vel} = \int_{t_0}^{t_f} \sqrt{\ddot{x}^2 + \ddot{y}^2 + \ddot{z}^2} dt \quad (5)$$

to represent the arclength along the velocity hodograph.

The deviation of a curve from a straight line is reflected by the curvature.²⁸ In the rotating frame of the CR3BP, curvature is calculated from the velocity and acceleration, $\mathbf{a} = [\ddot{x}, \ddot{y}, \ddot{z}]^T$, as²⁸

$$\kappa(t) = \frac{\|\mathbf{v} \times \mathbf{a}\|}{\|\mathbf{v}\|^3} = \frac{\sqrt{(\dot{x}\ddot{y} - \dot{y}\ddot{x})^2 + (\dot{z}\ddot{x} - \dot{x}\ddot{z})^2 + (\dot{y}\ddot{z} - \dot{z}\ddot{y})^2}}{(\dot{x}^2 + \dot{y}^2 + \dot{z}^2)^{3/2}} \quad (6)$$

States at a local maximum in curvature identify locations where the trajectory bends the most.²⁸

The Frenet frame is defined by a set of vectors that evolve with the shape of a trajectory. The frame is defined by a tangent vector, $\hat{\mathbf{V}}$; a principal normal vector, $\hat{\mathbf{N}}$; and a binormal vector, $\hat{\mathbf{B}}$. The tangent vector is aligned with the velocity vector, $\hat{\mathbf{V}} = \mathbf{v}/\|\mathbf{v}\|$. The binormal vector is normal to the osculating plane, equal to $\hat{\mathbf{B}} = \mathbf{v} \times \mathbf{a}/\|\mathbf{v} \times \mathbf{a}\|$. Finally, the principal normal vector points towards the center of curvature and is calculated as $\hat{\mathbf{N}} = \hat{\mathbf{B}} \times \hat{\mathbf{T}}$.²⁹

Volume Representations

Several algorithms exist to represent the volume that encompasses data. In this paper, voxels, three-dimensional cube analogs to pixels,³⁰ are used as a discrete representation of a continuous volume that encompasses the data. In addition to representing where data exists in an environment, voxels can also encode metadata describing selected characteristics.³⁰

Conflict Based Search

The conflict-based search algorithm³¹ is used in path planning to generate conflict-free paths of multiple agents in an environment. The conflict-based search creates and searches a conflict tree for a set of optimal conflict-free paths. Each node of the tree contains a set of optimal paths with one for each agent and an overall cost for all agents.³¹ Within a node, optimal paths for each agent are identified individually, without considering the locations of the other agents.³¹ Then, if two or more agent paths represented by a node are in conflict, the node splits and produces one child node for each of the agents in conflict.³¹ In each child node, one of the conflicting agents is assigned a path constraint to avoid the conflict of the parent.³¹ Each child node possesses either the same or a higher cost than the parent.³¹ The algorithm continues to expand the tree down from the root until conflict-free nodes are created.³¹ The algorithm then returns the set of agent paths from the conflict-free node with the lowest cost.

Library of Motion Primitives and Their Regions of Existence

A library of motion primitives is used as an input to this paper. The motion primitives provide fundamental representations of motion within the system while their regions of existence describe the volume for which they represent. This library was previously generated by Gillespie, Bosanac, Bodin, and Miceli³² with subrepresentatives following the approach of Bodin, Bosanac, and Gillespie¹⁸ as described later. Accordingly, only a brief, high-level overview of its definition and formulation is provided here.

The library of motion primitives summarizes a set of building-block sized arcs that correspond to motion that naturally approaches or departs unstable periodic orbits in the Earth-Moon CR3BP. A set of reference trajectories are propagated along the stable and unstable manifolds of selected periodic orbits. Each trajectory is propagated for a duration that is equal to three months plus a linear estimate of the time for a perturbation in the stable or unstable eigenspace to grow by one order of magnitude. These trajectories are terminated early upon 1) impacting the Earth, modeled as a sphere with a radius of 6,378.137 km; 2) impacting the Moon, modeled as a sphere with a radius of 1,738 km; or 3) exceeding a nondimensional distance of unity from the Moon. Each trajectory is then discretized into smaller, overlapping arcs that are defined to span up to five curvature maxima and/or initial or final states, with four segments between them. Between two subsequent reference states, three sections of equal arc length are defined. Totaling up to thirteen per arc, these samples, along with the curvature maxima and/or initial or final states, describe the continuous arc at discrete and geometrically meaningful points. Each arc is described by a set of feature vectors that captures its shape. Groups of arcs exhibiting geometrically similar motion are then identified using a trajectory clustering algorithm, modeled after a convoy detection scheme. The medoid of each cluster, calculated by comparing the position vectors at all sampled states along the arcs, supplies the motion primitive.

In this paper, the motion primitive library contains 8,595 entries. Table 1 lists the number of primitives associated with arcs that approach or depart members along each family with Jacobi constants above 3.0. These arcs span both planar and spatial motions, but do not exhaustively summarize the solution space near the Moon.

Each cluster of arcs that resembles a motion primitive spans a volume in the phase space. As defined by Smith and Bosanac,³³ regions of existence approximate the volume that encompasses all trajectories within the cluster. Howell et al.³⁴ use “cells” to approximate hyperbolic invariant manifold surfaces. Similarly, this work leverages voxels to describe the regions of existence following a procedure demonstrated by Gillespie, Bosanac, Bodin, and Miceli.³² In addition to describing

Table 1. Number of motion primitives in the library, approaching or departing members of a specified periodic orbit family

Family	Number of Motion Primitives
L_1 Lyapunov	1268
L_1 northern halo	2041
L_1 southern halo	2041
L_2 Lyapunov	935
L_2 northern halo	1155
L_2 southern halo	1155

the location and shape of regions in the configuration space, these voxels also include additional metadata that encodes further properties of the trajectories that pass through them.

Each region is composed of a set of hierarchical voxels with each three-dimensional position voxel containing a set of three-dimensional velocity voxels that describe the velocities of trajectories passing through the position voxel. Voxels are described by their center, c_{pos} or c_{vel} in position or velocity space and their side length, l_{pos} or l_{vel} . Voxels are associated with sections of the region of existence, defined from the sections of the trajectory arcs between two subsequent feature vector samples. Voxel sizes may vary over a region, but remain consistent within a section. Elapsed time ranges from the start of the region are associated with each section and each voxel contains metadata associating it with a subrepresentative within the region, discussed in a subsequent paragraph.

Voxels are constructed around a high-resolution set of sampled states along the trajectories, known as a point cloud. The sampling frequency is defined to meet a user-specified tolerance of missed voxel detection events, characterized by a trajectory entering into the region of a voxel without a sample along it registered within the voxel bounds.¹⁸ In this work, samples are spaced evenly in the position-based arclength every $0.05l_{pos}^{min}$ as well as every $0.05l_{vel}^{des}$ in the velocity-based arclength where the superscript *min* indicates the minimum voxel position voxel size, 0.01 non-dimensional length units, and the superscript *des* represents the desired velocity voxel size as described in a subsequent paragraph.

Position voxels are scaled to better represent parts of the region that may be insufficiently sampled to fully capture the range of motion present within the cluster. As the density of trajectory arcs in clusters of the training dataset varies, some clusters may contain arcs that exceed a distance of l_{pos}^{min} from each other. Despite the fine sample frequency along trajectories, gaps may form between trajectories for these less dense clusters. Once a voxelized region is constructed, each section is evaluated by the following criteria: 1) the third largest distance between sampled points at either of the section boundaries must be less than the current position voxel size and 2) the longest arc length within the section must be less than a user-specified multiple of the voxel size. If either of these two criteria are not met, the position voxel size is increased to either $3 \times l_{pos}^{min}$ or $9 \times l_{pos}^{min}$. Scaling by factors of three ensures that voxel centers remain on the same grid defined by the centers of l_{pos}^{min} . Limiting the total growth to $9 \times l_{pos}^{min}$ bounds the loss of resolution and ensures that the voxels remain a realistic approximation of the data contained within them.

Unlike position voxels, the desired size of the velocity voxels is determined as the regions are constructed. Curves in the velocity hodograph tend to traverse a larger nondimensional distance than their corresponding curves in the configuration space, especially in dynamically sensitive regions like those near the Moon. As a consequence, many more minimum-size velocity voxels are required to fully capture the shape, contributing to larger data volumes and higher computational requirements when comparing regions. Considering the purpose of velocity voxels is to support ΔV estimates when transitioning between regions, it is less critical to fill velocity gaps and maintain a high resolution than in the configuration space. Furthermore, positioning of velocity voxels on a grid is not required. Consequently, velocity voxel sizes may grow unbounded to any integer multiple of $l_{vel}^{min} = 0.01$ non-dimensional velocity units. The value of l_{vel}^{des} is selected such that approximately ten velocity voxels span a section, balancing resolution with computational burden.

In addition to the voxels and section metadata, each region is characterized by a number of representative arcs. These representatives are selected using k -medoids clustering applied to their position-based feature vector spaces f_p evaluated along the entire trajectory. The number of medoids

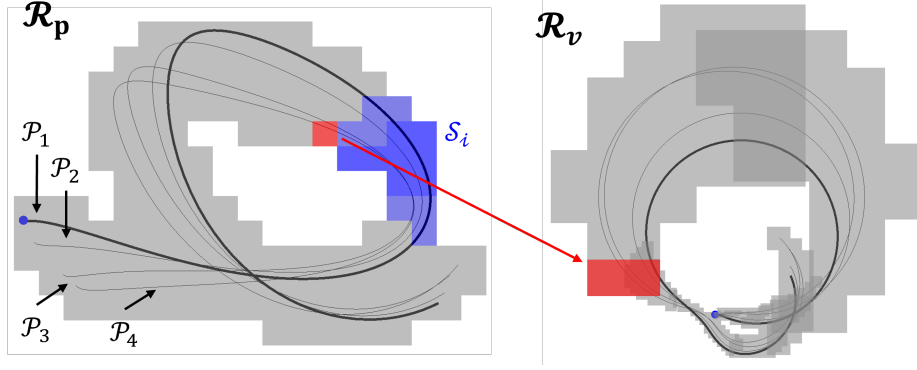


Figure 1. Position voxels (left), velocity voxels (right), and their representatives

is selected such that there is approximately one representative for every three position voxels across the region's widest section. Each position voxel is associated with its nearest representative or representatives. These representatives support initial guess construction with trajectory arcs that are representative of geometrically similar arcs near the voxels of interest.

An example region is shown in Figure 1. Position voxels $\mathcal{R}_p$ are shown on the left with section $\mathcal{S}_i$ highlighted blue. The red position voxel within $\mathcal{S}_i$ is associated with red velocity voxels, $\mathcal{R}_v$. Representatives $\mathcal{P}_1$ through $\mathcal{P}_4$ are shown in both position and velocity.

TECHNICAL APPROACH

The library of motion primitives and their associated regions of existence are used to construct trajectories of each agent in a DSM. In this work, a start and goal region are selected for each spacecraft, or agent, such that the agent's path begins in the first section of the start region and ends in the final section of the goal region. Between the boundary conditions, the initial guess algorithm identifies sequences of motion primitives, via their overlapping regions of existence in the configuration space, for each agent free of conflicts between other agents.

To demonstrate the technical approach, consider a scenario where two agents are operating near the Moon. The first agent starts from a trajectory that departs an L_1 Lyapunov orbit with periods $P \in [2.6917, 2.7698]$ and Jacobi constant $C_J \in [3.1675, 3.1883]$ and ends on a trajectory that is approaching an L_2 Lyapunov orbit with $P \in [3.3952, 3.4160]$ and $C_J \in [3.1519, 3.1612]$. At the same time, agent 2 starts on a trajectory that is departing an L_1 southern halo orbit with $P \in [2.6720, 2.7796]$ and $C_J \in [3.0221, 3.0664]$ and ends on a trajectory that approaches an L_2 northern halo orbit with $P \in [3.1637, 3.2373]$ and $C_J \in [3.0664, 3.0835]$. The start and goal regions for each agent are shown in Figure 2a) and b), respectively. Each agent starts in section one of the start region, colored in cyan on the left of each subfigure, and must reach section twelve of the goal region, colored in magenta on the right of each subfigure.

Creating Initial Guess Paths

Initial guesses for the trajectories of these two agents in a DSM are constructed by iteratively implementing three steps. First, an initial guess for each agent's individual trajectory is constructed using a multi-level graph search method developed by Bodin, Bosanac, and Gillespie.¹⁸ Next,

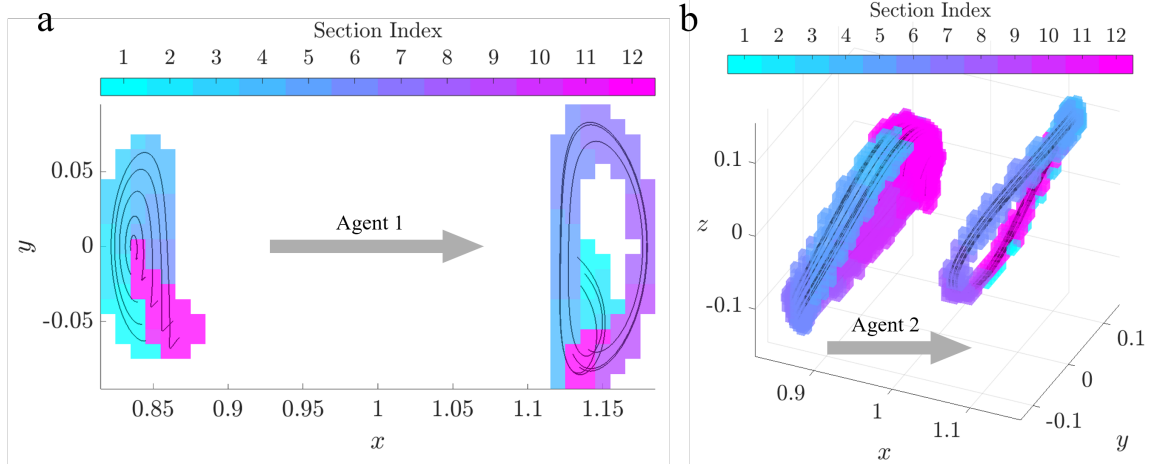


Figure 2. Start and goal regions for agent 1 (left) and agent 2 (right)

initial guesses for each agent are compared with each other for conflicts. Finally, constraints on each conflicting agent are applied as in the conflict-based search algorithm and the process repeats.

Single-Agent Initial Guess Construction At each level in the conflict-based structure, each agent must identify an initial guess for a path that minimizes the ΔV . To encode the transitions between subsequent primitives and their regions of existence as well as support accurate and efficient searches, a hierarchical graph structure is employed. This graph structure follows the process developed by Smith and Bosanac,¹⁹ refined by Miceli and Bosanac,²⁰ and combined into a hierarchy by Bosanac.³⁵ In this work, the layers of the hierarchy are each represented by a graph of distinct fidelity, defined as follows:

1. High-level region graph: connecting sequentially composable regions of existence
2. Low-level section sequence graph: connecting sequentially composable sections within region-to-region connections

These two graphs are searched hierarchically to produce a sequence of sections that is processed to form an initial guess. First, the high-level region graph is searched to produce a candidate sequence of motion primitives with overlapping regions of existence. Then, the low-level section sequence graph is searched to determine if the sequence of identified regions can be traversed in forward time. The section sequences are then refined to ensure composability of arcs through the sections. However, if a viable initial guess is not generated, these steps are repeated with new paths through the high-level region graph until one is identified or a maximum iteration limit is reached.

To support generating both graphs, connections between sections along a pair of regions of existence is precomputed and stored. As edge weight calculation carries the highest computational burden of any step, this preprocessing step supports much faster initial guess generation.¹⁸ First, a pair of regions of existence within the library are compared to identify whether any of their position voxels intersect. In that case, their velocity voxels are compared to approximate the ΔV required to connect the two geometrically distinct types of motion at their intersection in the configuration space. To further reduce computational burden, an approximation of the velocity voxels for all intersecting position voxels within a section is employed using an axis-aligned bounding box (AABB) that encompasses them. The ΔV is then approximated as the shortest distance between the bound-

ing boxes of the two intersecting sections. If the distance is greater than 0.001 non-dimensional velocity units and the angle between the two velocities is greater than 30° , the velocity change is too large to be considered a useful connection.

The high-level region graph is defined following the formulation presented by Bodin, Bosanac, and Gillespie.¹⁸ Specifically, the nodes correspond to regions within the library and undirected, weighted edges indicate the cost of connecting two intersecting regions. Searching this graph produces a sequence of regions of existence that overlap in the configuration space. However, it is necessary to determine whether a spacecraft can traverse a sequence of sections along each region in forward time.

For a sequence of overlapping regions of existence, a subgraph is created to find the lowest cost path through the sections of each region. This low-level graph is constructed with nodes representing sections and edges weighted by the ΔV to transfer from one section to the next. However, edges between sections of the same region are defined with a weight of zero. Paths in this graph determine when and where an agent should transition from one region in the higher level graph to the next.

Each graph is searched to produce one or more paths that connect the selected start and end nodes and minimize the total ΔV . In this paper, Yen’s algorithm³⁶ is used to find a user-defined number of paths, sorted in increasing order of cost as defined by the sum of edge weights along the path. Yen’s algorithm finds the k shortest paths in a graph using Dijkstra’s algorithm on deviations around the optimal path.³⁶ This graph search algorithm has previously been used in astrodynamics problems by Bruchko and Bosanac³⁷ as well as Miceli and Bosanac.²⁰

Path refinement, developed by Miceli and Bosanac²⁰ and extended by Bodin, Bosanac, and Gillespie,¹⁸ selects individual trajectory arcs from the sequence of sections that form the components of the initial guess while ensuring composability between them. Candidate arcs are selected from the representatives of a region that span the sequence of sections defined by the paths of the previous step. These candidate arcs supply nodes of a refinement graph, the lowest-level graph of this process, and are sampled at the initial and final state of each section. In this graph, nodes are connected if any intersecting voxels are associated with the representatives in both regions and the velocities of samples bounding the sections meet the same criteria of the velocities in the region-level graph connections. Edge weights, w , are defined from the position discontinuity between two samples along the arcs, $\Delta \mathbf{r}$, and the angle between the velocity vectors, θ , as $w = c_p |\Delta \mathbf{r}| + c_v (1 - \cos \theta)$ where c_p and c_v are weighting coefficients for position and velocity set to 10 and 1 respectively. The graph search is performed using a modified A^* search presented by Miceli and Bosanac,²⁰ producing the sequence of arcs that supply the initial guess for the agent.

There is a possibility that the refinement process produces no viable initial guesses for a given region and section sequence. In the case that a path is not found, the process repeats, returning to the high-level motion primitive graph and searching for additional paths via Yen’s algorithm. To reduce computational burden, batches of ten region paths are created at a time and input to the refinement process. Should any path successfully be refined, the algorithm terminates, returning the initial guess for the agent. Otherwise, the algorithm continues until a maximum number of paths that fail to refine is reached.

Multi-Agent Path Constraint Evaluation Conflicts are any user-defined constraint that the initial guess paths for each agent must follow. In this paper, conflicts are defined to prevent any number of agents from being in the same place at the same time. Two formulations of this constraint are implemented to provide options regarding how conservatively the constraint should be considered.

The first implementation prevents two agents from occupying the same region and the same section at the same time. The second, more conservative, approach checks that two agents can not occupy two intersecting sections of any regions at the same time.

Time compatibility, i.e., ensuring that two agents are not in conflicting locations at the same time, is checked by comparing ranges of possible elapsed times, regardless of the approach used to determine intersection. Let ${}_i\mathcal{T}_k$ represent the possible elapsed time ranges from the start of region i to section k . Lower bounds of the time ranges are represented by ${}_i\mathcal{T}_k^-$ with upper bounds represented by ${}_i\mathcal{T}_k^+$. Possible transit times for a spacecraft that enters region i at section n and departs the region at section $m > n$ are constrained by lower bounds ${}_i\mathcal{T}^{-'} = {}_i\mathcal{T}_m^- \ominus {}_i\mathcal{T}_n^+$ and upper bounds ${}_i\mathcal{T}^{+'} = {}_i\mathcal{T}_m^+ \ominus {}_i\mathcal{T}_n^-$. If the spacecraft then continues into region j , similar ${}_j\mathcal{T}^{-'}$ and ${}_j\mathcal{T}^{+'}$ bounds are associated with the possible times within this region. The total possible times to transit both regions i and j are bounded by $\mathcal{T}^{-''} = {}_i\mathcal{T}^{-'} \oplus {}_j\mathcal{T}^{-'}$ below and $\mathcal{T}^{+''} = {}_i\mathcal{T}^{+'} \oplus {}_j\mathcal{T}^{+'}$ above. Possible elapsed time ranges are accumulated for each agent in this manner for each region that it passes through. If two agents are deemed to be in the same location at sufficiently similar times, then the time ranges, ${}^\alpha\mathcal{T}$ and ${}^\beta\mathcal{T}$, for agents α and β in their respective sections of intersection are compared through differences in their possible time ranges with lower bounds $\mathcal{L} = {}^\beta\mathcal{T}^- \ominus {}^\alpha\mathcal{T}^+$ and upper bounds $\mathcal{U} = {}^\beta\mathcal{T}^+ \ominus {}^\alpha\mathcal{T}^-$. If $\exists l \mid \mathcal{L}_l \leq 0 \leq \mathcal{U}_l : l = 1 \dots n(\mathcal{L}) = n(\mathcal{U})$, then a difference in elapsed time of zero is possible between the two agents within the section, meaning that they could be in the section at the same time.

Conflict-Based Tree Search If a conflict has been identified, alternative path options are generated and evaluated in a tree structure following a conflict-based search approach.³¹ In this work, a recursive approach is used to build a conflict tree. Starting from a single root node, each node in this tree includes a path for each agent as well as the total cost for all the paths in the node. If the paths within the node are in conflict, child nodes are created where each child is assigned a different constraint to avoid the conflict.

In the example case described earlier, the section-intersection formulation of the constraint is used. Consider two paths for agent 1 traversing between the regions of Figure 2a) and agent 2 traversing between the regions of Figure 2b). The root node selects a path for agent 1, A1, that traverses from its start region #1247 through region #629 to its goal region of #5592 while A2 starts in #3472, transitions to #8384, and reaches its goal of #7026. However, a conflict is identified between regions #629 in A1 and #8384 in A2 because some voxels are intersecting between them and it is possible for the two agents to be in intersecting sections at the same time. For instance, agent 1 could be in section 2 of region #629 anywhere between 0 and 2.29 non-dimensional time units after the epoch. Additionally, agent 2 could be in section 8 of region #8384 anywhere between 2.11 and 5.26 non-dimensional time units of the epoch. Because these two sections contain intersecting voxels and there is a possibility that the two agents are in the respective section at the same time (anywhere between 2.11 and 2.29 units of elapsed time), a time conflict is conservatively deemed to occur.

A set of forbidden regions of existence for each agent is tracked through the nodes of the conflict tree. When a child node is created, the conflicting regions, excluding start or goal regions, for the respective agent are added to the list of forbidden nodes. Before searching the graph for the agent in conflict, the weight of all edges leading into or out of forbidden nodes are set to infinity. After searching the graph, the edge weights are reset to ensure that nodes of other branches are not affected. In a node, if no new constraints are added to any particular agent, the agent path is inherited from the parent to reduce redundant individual agent graph searches.

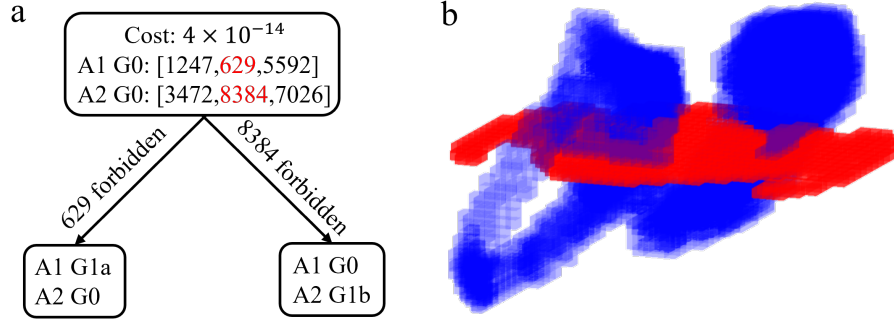


Figure 3. Conflict tree (left) grown from conflicting regions (right)

Figure 3 demonstrates a) growth from the root node with the conflict tree and b) the initial guess region sequence represented in the root. Consistent with the example of a potential intersection from the previous subsection, there is a conflict between regions #629 of A1 and #8384 of A2, prompting a split of the root node into two child nodes. The leftmost child node is assigned a new constraint that region #629 is forbidden from the A1 path. A new graph for agent 1, G1a, is constructed with all incoming and outgoing edges of region #629 are set to infinity. In this node, the A2 path remains the same as the root and the graph for agent 2 is not modified. In the rightmost child node, the other side of the conflict is explored by forbidding A2 from containing region #8384 and leaving A1 unchanged. Within each child node, new paths are found through the updated graphs. Then, the resulting paths are checked for conflicts with each other.

The conflict tree is grown until a conflict-free path is identified or a user-defined maximum recursion depth is reached. Because each individual agent path represents the minimum cost within its graph and the node cost is the sum of individual agent costs, lower cost DSM initial guesses are higher up on the tree. Paths that contain conflicts are stored and returned along with the solution path, if it exists before the recursion depth is reached. It is still possible, though somewhat less likely, that these paths may still be correctable in the subsequent step, especially those rejected by the more conservative constraint evaluation approach.

Correcting Initial Guesses

Once the initial guesses have been constructed, they are corrected for each agent individually using a multiple-shooting approach.³⁸ Using the same formulation as Bodin, Bosanac, and Gillespie,¹⁸ the representative arcs selected from each region are segmented at section boundaries from each region. Additionally, these arcs are also segmented based on perilune passages.¹⁸ Thus, each initial guess is segmented into a sequence of n_{arc} smaller arcs. A free variable vector is constructed from the initial states, $\mathbf{X}$, along each of these smaller arcs along with the propagation time of the arc, T . This vector is defined as:¹⁸

$$\mathbf{V} = [\mathbf{X}_1^T, T_1, \mathbf{X}_2^T, T_2, \dots, \mathbf{X}_{n_{arc}}^T, T_{n_{arc}}, \beta_1, \beta_2, \dots, \beta_{n_{arc}}]^T$$

Slack variables, β , are included at the end of the free-variable vector and used to enforce a non-negative propagation time along each arc and are initialized as $\beta_i = \sqrt{T_i}$. The i th arc, described by this free variable vector, is also used to generate the associated state transition matrix, Φ_i .

In the corrections scheme, position continuity is enforced between all arcs in the multiple-shooting

problem whereas velocity continuity is only enforced between arcs where an impulsive maneuver is not allowed. Maneuver locations between subsequent arcs are represented by a logical vector, $\mathbf{b}$, with a value of b_i indicating if the i^{th} node of $\mathbf{V}$ occurs immediately after a maneuver. Furthermore, the initial state along a trajectory is constrained to equal a desired value of $\mathbf{X}_0$. However, the final state is only constrained in position to match a desired value $\mathbf{r}_{goal}$, allowing for a final impulsive maneuver to match the velocity of a desired state vector. Thus, the constraint vector is defined as:

$$\mathbf{F} = \begin{bmatrix} \mathbf{X}_0 - \mathbf{X}_i^0 \\ \mathbf{F}_1 \\ \vdots \\ \mathbf{F}_{n_{arc}-1} \\ \mathbf{r}_{n_{arc}}^f - \mathbf{r}_{goal} \\ T_1 - (\beta_1)^2 \\ \vdots \\ T_{n_{arc}} - (\beta_{n_{arc}})^2 \end{bmatrix} \quad \mathbf{F}_i = \begin{cases} \mathbf{r}_i^f - \mathbf{r}_{i+1}^0 & \text{if } b_i = 1 \\ \mathbf{X}_i^f - \mathbf{X}_{i+1}^0 & \text{if } b_i = 0 \end{cases}$$

where a superscript 0 indicates the initial state of an arc and a superscript f represents the propagated final state of the arc. The gradient of the constraint vector, $\mathbf{F}$, with respect to the free variable vector, $\mathbf{V}$, is expressed as the following DF matrix:

$$\mathbf{DF} = \begin{bmatrix} -\mathbf{I}_{6 \times 6} & \mathbf{0}_{6 \times 14n_{arc}+1} \\ & \mathbf{M}_1 \\ & \vdots \\ & \mathbf{M}_{n_{arc}-1} \\ [0_{3 \times 13n_{arc}}, \Phi^{rr}_{n_{arc}}, \Phi^{rv}_{n_{arc}}, \mathbf{v}_{n_{arc}}, 0_{3 \times n_{arc}}] & \\ & \mathbf{B}_1 \\ & \vdots \\ & \mathbf{B}_{n_{arc}} \end{bmatrix} \quad \Phi = \begin{bmatrix} \Phi^{rr} & \Phi^{rv} \\ \Phi^{vr} & \Phi^{vv} \end{bmatrix}$$

$$\mathbf{M}_i = \begin{cases} [0_{3 \times 7(i-1)}, \Phi^{rr}_i, \Phi^{rv}_i, \mathbf{v}_i, -\mathbf{I}_{3 \times 3}, 0_{3 \times 14n_{arc}-7(i-1)}] & \text{if } b_i = 1 \\ [0_{6 \times 7(i-1)}, \Phi_i, \dot{\mathbf{X}}_i, -\mathbf{I}_{6 \times 6}, 0_{6 \times 14n_{arc}-7(i-1)}] & \text{if } b_i = 0 \end{cases}$$

$$\mathbf{B}_i = [0_{1 \times 7i-1}, 1, 0_{1 \times 13n_{arc}+i-1}, -2\beta_i, 0_{1 \times n_{arc}-i}]$$

This matrix possesses a dimension of $(6 + 6n_{coast} + 3n_{burn} + 3 + n_{arc}) \times (7n_{arc})$ where n_{coast} is the number of natural segments and n_{burn} is the number of segments with a maneuver.

The free variable vector is updated using Newton's method. The update equation is defined as $\mathbf{V}^{j+1} = \mathbf{V}^j - \mathbf{DF}^T [\mathbf{DF} \mathbf{DF}^T]^{-1} \mathbf{F}^j$ for each iteration j . The iteration continues until $|\mathbf{F}^j| < \epsilon_{corr}$ where ϵ_{corr} is a user-defined tolerance, set to 1×10^{-10} in this work. The resulting free variable vector provides states along the continuous corrected agent trajectory.

Resulting paths from the example above using the less-conservative constraint criteria are individually corrected for each agent and shown in Figure 4. The blue trajectory shows the path after corrections with the initial guess plotted using unique colors for each arc. Maneuver locations are circled in black. Note that two maneuvers occur in close succession at the end of the path for agent 1. After corrections, the maneuver magnitudes are 49.6547, 40.2970, 9.7707, 20.1084, 139.2805, 7.9998, 2.1200, and 0.2131 m/s, respectively, for agent 1 and 192.0648, 91.4334, 115.6365, and

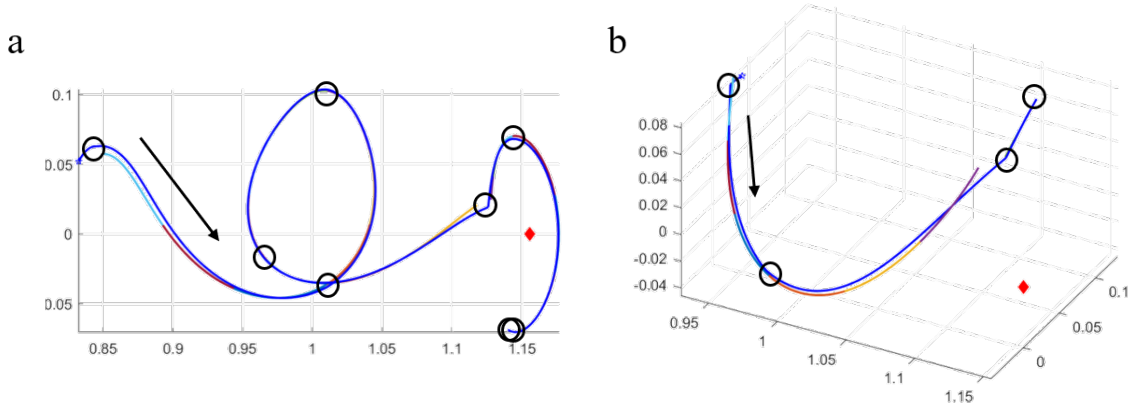


Figure 4. Individually corrected paths for a) agent 1 and b) agent 2

42.2644 m/s for agent 2. Agent 1 traverses its corrected path in 22.46 days whereas agent 2 requires a flight time of 4.68 days. In these examples, the corrected trajectory retains the geometry of the initial guess.

RESULTS

The presented technical approach is applied to several example DSM configurations to produce continuous, corrected paths for each individual agent. The result is a set of paths constructed from conflict-free initial guesses that can be corrected together as a full DSM system in future work. These individually corrected agent paths reduce the likelihood of failure for a corrections scheme that operates over all agents at once and de-conflicts them from each other, planned for future work. This section demonstrates the capability to produce these initial guesses for multiple agents.

Two-Agent Example

Example 1: Recall the example from the technical approach that considers two agents. Agent 1 departs an L_1 Lyapunov orbit to approach an L_2 Lyapunov orbit whereas agent 2 departs an L_1 southern halo orbit and approaches an L_2 northern halo orbit. Both agents start at the specified initial states at the same time. When the initial guess is constructed using the less conservative constraint formulation in which two agents may not occupy the same section of the same region at the same time, agent 1 expends 269.4 m/s of ΔV during its 22.46-day transit time and agent 2 expends 441.4 m/s of ΔV during its 4.68-day transit. The two agents follow the paths shown in Figure 5 with arrows indicating their directions of motion and the Moon, L_1 , and L_2 shown in gray. Maneuver locations and magnitudes follow those described in the technical approach and displayed in Figure 4.

Example 2: If, however, the more conservative conflict definition is used with agents located in intersecting sections (regardless of their region) at the same time are considered in conflict, then the initial guess from the previous result is considered conflicting. In this case, the algorithm continued until it reached its maximum recursion depth of one for this example. While a conflict-free initial guess was not found at that level, an initial guess was still returned since there is still a possibility of correction. Indeed, the individual agent paths were successfully refined. The path of agent 1 remains unchanged from the Example 1, however, the path for agent 2 now expends 1.5534 km/s of

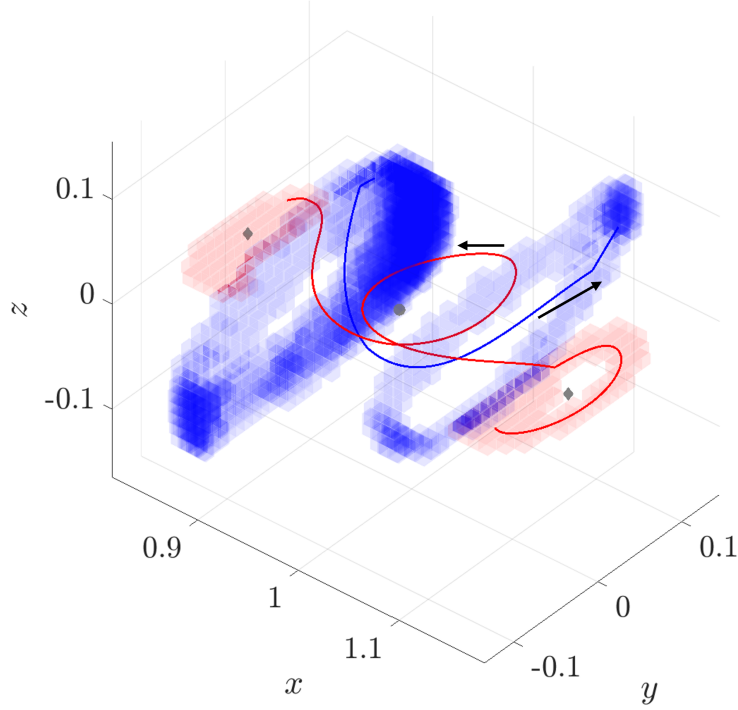


Figure 5. Agent 1 (red) and Agent 2 (blue) following their individually corrected paths for a successful initial guess

ΔV during its 11.62-day transit. The paths for these agents are shown in Figure 6. Maneuvers along the new path of agent 2 are indicated using black circles with magnitudes of 532.9587, 204.3955, 578.4002, 70.0251, 63.7899, 86.0360, 11.7235, 5.6743, and 0.4140 m/s.

Three-Agent Example

Example 3: The same technical approach is applicable to any number of spacecraft within a DSM. To demonstrate this procedure, a third agent is added to the scenario from Example 1. This agent starts in a region departing an L_2 Lyapunov orbit with $P \in [3.3736, 3.4160]$ and $C_J \in [3.1519, 3.1720]$ and must find a path that eventually approaches an L_1 Lyapunov orbit with $P \in [2.7421, 2.8237]$ and $C_J \in [3.1547, 3.1746]$. For demonstration purposes, this scenario uses the constraint forbidding agents from being in the same region at the same time. The resulting corrected individual agent paths are shown in Figure 7. From the root node, the solution recovers the same paths for agent 1 and agent 2 as Example 1 with agent 3 traversing its path in 30.78 days and expending a total 532.8 m/s of ΔV . This total ΔV is composed of individual maneuver magnitudes of 6.6176, 21.1261, 92.6266, 17.9860, 1.2424, 19.3155, 368.2181, 2.7839, 2.0655, 0.5928, 0.2085, and 0.0539 m/s, with locations indicated by black circles in Figure 7.

CONCLUSIONS

This paper extends prior work in trajectory design using motion primitives to construct initial guesses for distributed spacecraft missions and correct the resulting individual agent paths. This paper uses as an input a library of motion primitives and their regions of existence to summarize a subset of trajectories that exist near the Moon and approach or depart selected unstable periodic

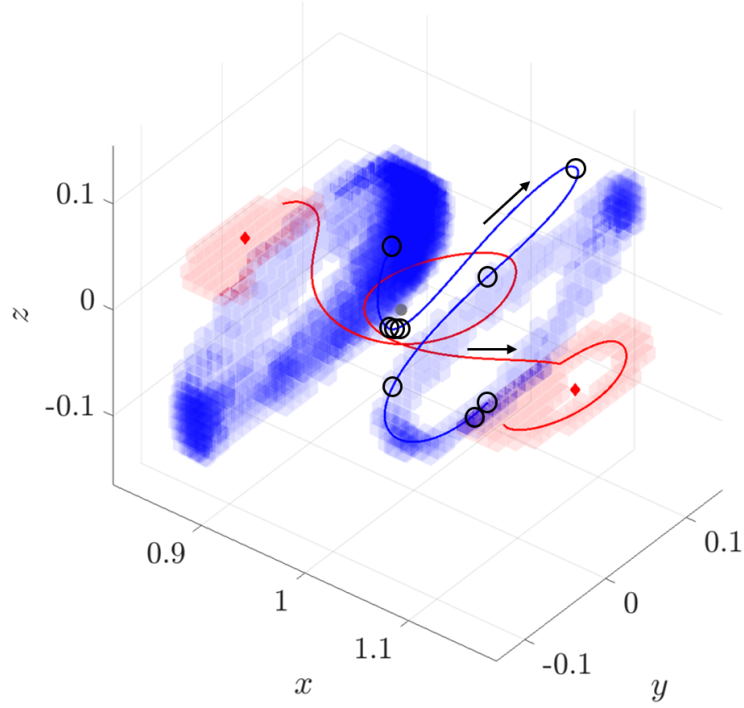


Figure 6. Agent 1 (red) and Agent 2 (blue) following their individually corrected paths for an unsuccessful initial guess

orbits. The regions of existence are used to rapidly generate initial guesses for each individual agent. Sequences through intersecting regions along with transition sections are identified through a multi-level graph search. The resulting section sequences for each agent are compared for conflicts and a conflict tree is grown following the principles of conflict-based search. If a conflict-free section sequence is found before a maximum recursion depth, it is returned as the initial guess candidate. Otherwise, a conflicting section sequence is selected as the initial guess candidate. Representative arcs from the initial guess supply multiple shooting arcs for each agent that are then corrected into continuous trajectories for all spacecraft within the DSM.

Future work includes correcting and optimizing the DSM trajectories together. During the individual agent correction process, it is possible that a constraint is violated, necessitating a secondary multiple shooting corrections of all the agents simultaneously. Next, these corrected DSM trajectories need to be optimized to reduce the cost of impulsive maneuvers between segments.

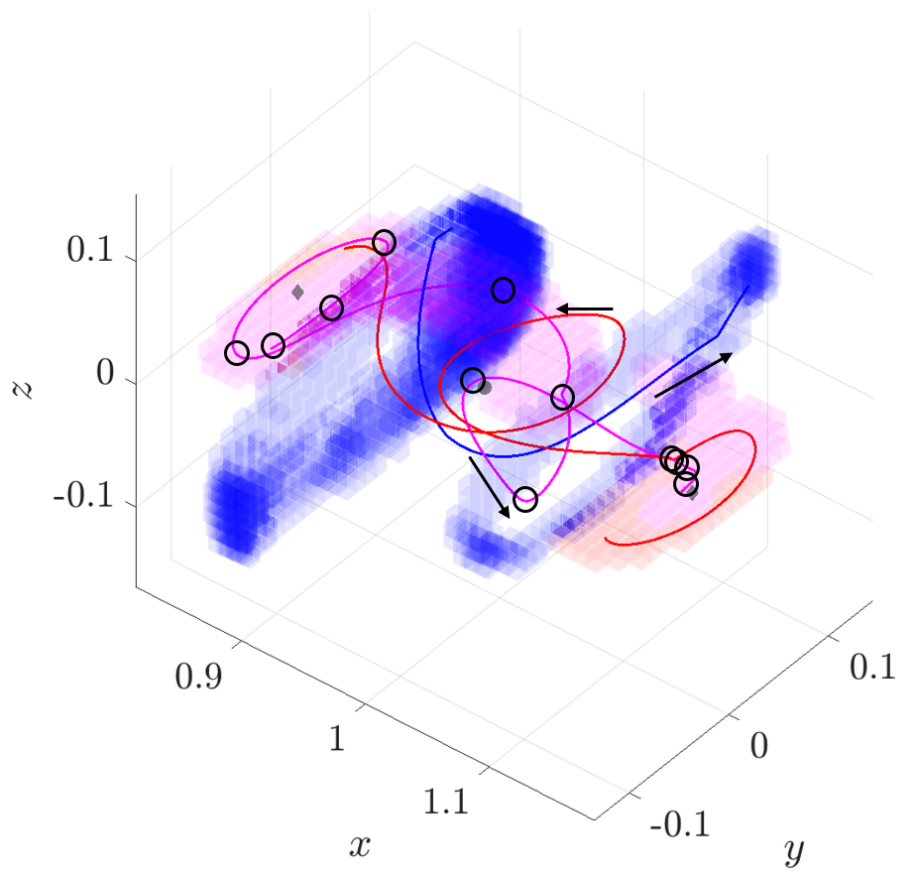


Figure 7. Agent 1 (red), Agent 2 (blue), and Agent 3 (magenta) following their individually corrected paths

ACKNOWLEDGMENT

This material leverages as an input prior work, including a library of motion primitives and code for generating initial guesses, that was performed under a project supported by the Air Force Office of Scientific Research under award number FA9550-23-1-0235. The new work presented in this paper was not supported by this award. The first author also thanks Austin Bodin for assistance with using prior code for single-spacecraft initial guess construction.

REFERENCES

- [1] J. Le Moigne, J. C. Adams, and S. Nag, “A new taxonomy for distributed spacecraft missions,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 13, 2020, pp. 872–883.
- [2] NASA, “NASA’s Lunar Exploration Program Overview,” *National Aeronautics and Space Administration*, 2020, pp. 1–75.
- [3] J. G. Walker, “Circular Orbit Patterns Providing Continuous Whole Earth Coverage,” Tech. Rep. RAE Technical Report 70211, Royal Aircraft Establishment, Farnborough, UK, Nov. 1970.
- [4] L. Rider, “Optimized Polar Orbit Constellations for Redundant Earth Coverage,” *Journal of the Astronautical Sciences*, Vol. 33, No. 2, 1985, pp. 147–161.
- [5] L. Rider, “Analytic Design of Satellite Constellations for Zonal Earth Coverage Using Inclined Circular Orbits,” *Journal of the Astronautical Sciences*, Vol. 34, 1986, pp. 31–64.
- [6] Y. Ulybyshev, “Satellite Constellation Design for Complex Coverage,” *Journal of Spacecraft and Rockets*, Vol. 45, No. 4, 2008, pp. 843–849.
- [7] D. Arnas and R. Linares, “Uniform satellite constellation reconfiguration,” *Journal of Guidance, Control, and Dynamics*, Vol. 45, No. 7, 2022, pp. 1241–1254.
- [8] P. Levinson-Muth, M. Joyner, and L. Plice, “HelioSwarm: Transfer Trajectory and P/2 Lunar Resonant Orbit Design for a Multi-Satellite Observatory,” *AIAA SCITECH 2024 Forum*, 2024, 10.2514/6.2024-1270.
- [9] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.
- [10] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of robot motion: theory, algorithms, and implementations*. MIT press, 2005.
- [11] R. Stern, “Multi-agent path finding—an overview,” *Artificial intelligence: 5th RAAI summer School, dolgoprudny, Russia, july 4–7, 2019, tutorial lectures*, 2019, pp. 96–115.
- [12] A. Wolek and C. A. Woolsey, *Model-Based Path Planning*. Cham: Springer International Publishing, 2017, 10.1007/978-3-319-55372-6_9.
- [13] 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*, Vol. abs/1405.5581, 2014.
- [14] 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, 10.1109/CVPRW50498.2020.00520.
- [15] A. Majumdar and R. Tedrake, “Funnel Libraries for Real-Time Robust Feedback Motion Planning,” *CoRR*, Vol. abs/1601.04037, 2016, <https://doi.org/10.1177/0278364917712421>.
- [16] T. R. Smith and N. Bosanac, “Constructing motion primitive sets to summarize periodic orbit families and hyperbolic invariant manifolds in a multi-body system,” *Celestial Mechanics and Dynamical Astronomy*, Vol. 134, No. 7, 2022, 10.1007/s10569-022-10063-x.
- [17] 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.
- [18] A. Bodin, N. Bosanac, and C. Gillespie, “Long-Term Spacecraft Trajectory Prediction Using Behavioral Motion Primitives,” *AAS/AIAA Astrodynamics Specialist Conference*, 2025.
- [19] 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, 10.1007/s40295-023-00395-7.
- [20] G. E. Miceli and N. Bosanac, “Generating Planar Trajectories for Neptunian System Exploration Using Motion Primitives,” *The Journal of the Astronautical Sciences*, Vol. 73, Feb 2026, p. 11, 10.1007/s40295-025-00545-z.
- [21] N. Bosanac, “Using Motion Primitives to Generate Initial Guesses for Trajectories to Sun-Earth L2,” *AAS/AIAA Astrodynamics Specialist Conference*, August 2025.

- [22] V. Szebehely, *Theory of Orbits: The Restricted Problem of Three Bodies*. London: Academic Press, 1967.
- [23] R. Park, W. Folkner, J. Williams, and D. Boggs, “The JPL Planetary and Lunar Ephemerides DE440 and DE441,” *The Astronomical Journal*, Vol. 161, No. 105, 2021, 10.3847/1538-3881/abd414.
- [24] M. Hénon, *Generating Families in the Restricted Three-Body Problem*. Berlin, Heidelberg: Springer, 1997.
- [25] L. Perko, *Nonlinear Systems: Global Theory*, pp. 181–314. New York, NY: Springer New York, 2001, 10.1007/978-1-4613-0003-8_3.
- [26] W. Koon, M. Lo, J. Marsden, and S. Ross, *Dynamical Systems, The Three-Body Problem, and Space Mission Design*. New York, USA: Springer, 2011.
- [27] K. Wardle, *Differential Geometry*. Mineola, NY: Dover Publications, Inc., 2008.
- [28] N. Patrikalakis, T. Maekawa, and W. Cho, *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer, Berlin, Heidelberg, 2009. E-book.
- [29] T. Banchoff and S. Lovett, *Differential Geometry of Curves and Surfaces, 2nd Edition*. Boca Raton, FL.: CRC Press, 2016.
- [30] A. Kaufman, D. Cohen, and R. Yagel, “Volume graphics,” *Computer*, Vol. 26, No. 7, 1993, pp. 51–64, 10.1109/MC.1993.274942.
- [31] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, “Conflict-based search for optimal multi-agent pathfinding,” *Artificial intelligence*, Vol. 219, 2015, pp. 40–66.
- [32] C. Gillespie, N. Bosanac, A. Bodin, and G. Miceli, “Motion Primitive Approach to Spacecraft Trajectory and Behavioral Prediction in the Earth-Moon System,” *In Preparation*, 2026.
- [33] T. R. Smith and N. Bosanac, “Constructing Motion Primitive Sets to Summarize Periodic Orbit Families and Hyperbolic Invariant Manifolds in a Multi-Body System,” *Celestial Mechanics and Dynamical Astronomy*, Vol. 134, No. 1, 2022, p. 7, 10.1007/s10569-022-10063-x.
- [34] 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, 10.1007/BF03256477.
- [35] N. Bosanac, “Using Motion Primitives to Generate Initial Guesses for Trajectories to Sun-Earth L2,” *AAS/AIAA Astrodynamics Specialist Conference*, Boston, MA, August 2025.
- [36] 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.
- [37] K. 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, 10.1007/s40295-025-00506-6.
- [38] D. D. Morrison, J. D. Riley, and J. F. Zancanaro, “Multiple shooting method for two-point boundary value problems,” *Commun. ACM*, Vol. 5, Dec. 1962, p. 613–614, 10.1145/355580.369128.