AAS 22-583

DESIGNING SPATIAL TRANSFERS IN MULTI-BODY SYSTEMS USING ROADMAP GENERATION

Kristen L. Bruchko[*] and Natasha Bosanac[†]

Rapidly constructing trajectories in multi-body systems currently relies heavily on a human-in-the-loop with sufficient understanding of the solution space. However, probabilistic roadmap generation offers an alternative approach to efficiently summarizing the solution space and rapidly constructing constrained trajectories. In this paper, a graph is constructed using roadmap generation in the Earth-Moon circular restricted three-body problem to summarize motion flowing between L_1 and L_2. Then, a graph search algorithm is used to repeatedly search the graph to construct spatial transfers between libration point orbits.

INTRODUCTION

Designing trajectories in the chaotic regimes of multi-body systems currently relies heavily on a human-in-the-loop. However, the process often requires advanced knowledge of the solution space and a sufficient preliminary solution. Furthermore, constructing a trajectory that adheres to multiple mission constraints may lead to a potentially time-consuming process that limits the rapid design of efficient solutions as the requirements or hardware parameters evolve. Path planning techniques, such as probabilistic roadmap generation, may offer an alternative approach that reduces the burden on a human-in-the-loop during the trajectory design process.

Path planning techniques focus on computing collision free paths subject to dynamical constraints in environments containing obstacles.[1] These techniques are commonly used in robotics, artificial intelligence, and control theory to efficiently summarize and plan paths, or trajectories, in complex environments.[2] Probabilistic roadmap generation is a sampling-based path planning technique that independently constructs a graph, labeled the roadmap, that sufficiently summarizes an environment by capturing the global dynamics and sensitive regions. The graph consists of nodes, which are sampled configurations in the environment, and edges, which are local paths between nodes. The nodes and edges added to the graph are considered to be valid, which means they adhere to any static or dynamical constraints. Once the roadmap is completed, it may be searched repeatedly for a variety of solutions from a start configuration(s) to a goal configuration(s) while also adhering to multiple constraints. A well-constructed roadmap enables the rapid design of solutions. This approach has been demonstrated to solve problems in high-dimensional spaces with probabilistic completeness, i.e., the probability of finding a solution if one exists approaches unity as the number of nodes increases.[3]

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

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

Previous studies have explored the application of path planning strategies to trajectory design. Automated path finding techniques have been demonstrated in astrodynamics by Das-Stuart, Howell, and Folta to recover an initial guess in a variety of transfer scenarios using Dijkstra's graph search method to search a database of fundamental solutions.[4] In addition, a tree-based search has been used by Starek et. al. to construct propellant-optimized solutions for rendezvous and proximity operations near circular orbits in a dynamical environment governed by the modeled by Clohessy-Wiltshire-Hill equations.[5] Trumbauer and Villac have also used a precomputed database of periodic orbits and invariant manifolds to generate a directed graph and design transfers between periodic orbits in lower-fidelity models using discrete search algorithms, which are then corrected and optimized during onboard trajectory redesign.[6] These previous applications of path planning methods to astrodynamics further motivate the exploration of roadmap generation for efficient construction of complex trajectories in multi-body systems.

In this paper, a roadmap is constructed in the Earth-Moon Circular Restricted Three Body Problem (CR3BP) to summarize the solution space near the vicinity of L_1 and L_2 across a range of energy levels. This work expands and improves on the approach presented in a previous paper that focused on a preliminary implementation for planar motion.[7] The nodes that form the roadmap are selected iteratively using both states that lie along periodic orbits and their stable and unstable manifolds as well as states that are dispersed through the broader phase space. Using states that lie along fundamental solutions that govern natural transport in the CR3BP partially biases the roadmap to capture some differences in sensitivity in distinct regions of the phase space. These nodes of the roadmap are then connected by edges to neighboring nodes. The edges are local paths that represent valid motion between nodes and are constructed using a combination of natural and maneuver-enabled arcs. The edges are weighted by the total maneuver magnitude required to traverse the trajectory connecting the parent node to the neighboring node.

After the roadmap is generated, a global search algorithm is used to search the roadmap multiple times to construct a variety of initial guesses for transfers between L_1 and L_2 periodic orbits. These transfers possess a variety of characteristics such as total maneuver requirements, specified constrained departure and arrival states, or constrained departure and arrival regions. A primary benefit of roadmap generation is the ability to reuse the same roadmap repeatedly to construct distinct transfers. The time the planner spends generating a sufficient roadmap is about 95% of the total preprocessing time. However, after this initial computation, the roadmap may then be searched rapidly for a variety of complex solutions.[8] In this paper, the roadmap is searched using Dijkstra's search algorithm which allows the graph to be searched with respect to a desired heuristic. This search produces an initial guess that is input to a corrections scheme to produce a continuous trajectory. By generating a single roadmap to summarize a portion of the complex dynamical environment near the vicinity of the Moon, the goal is efficiently and rapidly generating natural planar transfers or maneuver-enabled spatial transfers without heavily burdening a human-in-the-loop.

BACKGROUND: DYNAMICAL MODEL

The CR3BP models the motion of a spacecraft under the gravitational influence of two larger bodies, such as the Earth and the Moon. In this model, the spacecraft is assumed to have negligible mass compared to both primary bodies. The larger body, P_1, and the smaller body, P_2, are modeled as point masses with constant mass, M_1 and M_2, respectively. Both primaries are assumed to follow circular orbits about their mutual barycenter. Then, a rotating coordinate frame is defined with the origin at the barycenter of the system and axes $\hat{x}, \hat{y}, \hat{z}$: $\hat{x}$ is directed from P_1 to P_2, $\hat{z}$ is aligned with

the system's orbital angular momentum vector, and $\hat{y}$ completes the right-handed orthogonal triad.[9] The mass, distance, and time quantities are then nondimensionalized by the characteristic quantities m^*, l^*, and t^*: m^* is set equal to the total mass of the primaries, t^* is defined such that the mean motion of the primary bodies is unity, and l^* is set equal to the assumed constant distance between the primaries. The nondimensional state of the spacecraft is then written in the rotating frame as $\boldsymbol{x} = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^T$. Using these assumptions and definitions, the nondimensional equations of motion for a spacecraft in the CR3BP are:

$$\begin{aligned}
\ddot{x} &= 2\dot{y} + x - \frac{(1-\mu)(x+\mu)}{r_1^3} - \frac{\mu(x-1+\mu)}{r_2^3} \\
\ddot{y} &= -2\dot{x} + y - \frac{(1-\mu)y}{r_1^3} - \frac{\mu y}{r_2^3} \\
\ddot{z} &= -\frac{(1-\mu)z}{r_1^3} - \frac{\mu z}{r_2^3}
\end{aligned} \tag{1}$$

where $\mu = M_2/(M_1 + M_2)$ is the mass ratio of the system and the distances of the spacecraft from P_1 and P_2, respectively, are $r_1 = \sqrt{(x+\mu)^2 + y^2 + z^2}$ and $r_2 = \sqrt{(x-1+\mu)^2 + y^2 + z^2}$. In this dynamical model, the Jacobi constant is an integral of motion that equals

$$C_J = (x^2 + y^2) + \frac{2(1-\mu)}{r_1} + \frac{2\mu}{r_2} - \dot{x}^2 - \dot{y}^2 - \dot{z}^2 \tag{2}$$

At a single value of the Jacobi constant, the CR3BP admits fundamental solutions such as equilibrium points, periodic orbits, and invariant manifolds that exist in continuous families.[10] Unstable periodic orbits possess stable invariant manifolds that approach the periodic orbit and unstable invariant manifolds that depart the periodic orbit, as time tends to infinity. These invariant manifolds guide natural transport within the system.

BACKGROUND: NUMERICALLY CORRECTING TRAJECTORIES

To correct the discontinuous initial guesses for transfers constructed in this paper, a two-point boundary value problem is solved using a multiple-shooting corrections algorithm. In this approach, a trajectory is discretized into multiple arcs; the i-th arc is described by its initial state $\boldsymbol{x}_i$ and integration time Δt_i. Along with the final state along the final arc, the trajectory is described by n state vectors and $n-1$ integration times in the CR3BP. An initial guess for this trajectory is corrected to produce a continuous path using a free variable and constraint vector formulation of multiple-shooting. First, the states and integration times are assembled to form a $(7n-1)$-dimensional free variable vector $\boldsymbol{X}$ that is equal to:

$$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots, \boldsymbol{x}_n, \Delta t_1, \Delta t_2, \ldots, \Delta t_{n-1}]^T \tag{3}$$

Then, to recover a natural transfer, a $(6n-6)$-dimensional constraint vector $\boldsymbol{F}_n(\boldsymbol{X})$ reflects state continuity constraints between all neighboring arcs via the following definition:

$$\boldsymbol{F}_n(\boldsymbol{X}) = [\boldsymbol{x}_1(\Delta t_1) - \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n-1}(\Delta t_{n-1}) - \boldsymbol{x}_n]^T \tag{4}$$

where $\boldsymbol{x}_i(\Delta t_i)$ is the state produced by integrating the initial state $\boldsymbol{x}_i$ forward in time for Δt_i. Alternatively, to recover a maneuver-enabled transfer, only position continuity constraints are captured between the arcs where a maneuver is allowed; full state continuity is required between all remaining

neighboring arcs. If m maneuvers are allowed, these constraints form a $(6n-6-3m)$-dimensional constraint vector $\boldsymbol{F}_m(\boldsymbol{X})$. As an example, if a maneuver is located before the beginning of the $i+1$-th arc, $\boldsymbol{F}_m(\boldsymbol{X})$ takes the following form:

$$\boldsymbol{F} = [\boldsymbol{x}_1(\Delta t_1) - \boldsymbol{x}_2, \ldots, \boldsymbol{d}_i(\Delta t_i) - \boldsymbol{d}_{i+1}, \ldots, \boldsymbol{x}_{n-1}(\Delta t_{n-1}) - \boldsymbol{x}_n]^T \tag{5}$$

where $\boldsymbol{d}$ represents the position component of a state vector. Then, the free variable vector is iteratively updated from an initial guess by applying Newton's method until the norm of the constraint vector equals zero within a desired tolerance. The update equation at the i-th iteration is

$$\boldsymbol{X}_{i+1} = \boldsymbol{X}_i - D\boldsymbol{F}(\boldsymbol{X}_i)^T (D\boldsymbol{F}(\boldsymbol{X}_i) \cdot D\boldsymbol{F}(\boldsymbol{X}_i)^T)^{-1} \boldsymbol{F}(\boldsymbol{X}_i) \tag{6}$$

where $D\boldsymbol{F}(\boldsymbol{X})$ is the $(6n-6-3m) \times (7n-1)$ Jacobian matrix reflecting the partial derivatives of the constraints with respect to the free variables, calculated analytically in this paper.

BACKGROUND: PATH PLANNING

Path planning focuses on planning a sequence of valid paths between two points for an object within an environment. While the original purpose of path planning algorithms was to plan paths for robots, many industries have successfully adapted path planning algorithms to solve field-specific problems, such as engineers deciding if a part can be removed from an engine without complications or folding of peptides in molecular dynamics.[3] While in recent years autonomous path planning algorithms have been developed and adapted to solve more challenging problems such as navigation of changing environments and planning for robots with dynamical constraints, path planning algorithms in chaotic, nonlinear environments is still a challenge. One technique used in path planning is a sampling-based planning algorithm called probabilistic roadmap generation. This type of planner has shown high success planning in complicated high-dimensional solution spaces with obstacles and differential constraints, and have been adapted to solve a variety of problems.[2]

Probabilistic Roadmap Generation

Probabilistic Roadmap Generation (PRM) is a sampling-based planner that summarizes the environment by constructing a graph or roadmap which can be searched for a valid path between two configurations. A main benefit of PRM is that once a roadmap is constructed for an environment, it can be searched multiple times for a variety of configurations and constraints, rapidly producing a variety of valid paths based on user defined criteria. PRM is also a probabilistically complete planner: if a solution path exists, the planner will eventually find it as the number of nodes and edges in the graph grow.[3] Planning via PRM is typically divided into two phases: the learning phase and the query phase. The learning phase is where the graph is constructed and the query phase is where the roadmap is searched to find a valid path between a start and goal configuration.

During the learning phase, the roadmap is constructed by adding valid nodes and edges to the graph until it summarizes the environment. Nodes are sampled configurations within the solution space that are valid, meaning they adhere to all dynamic and static constraints. The sampling scheme used to select nodes for a specific problem can vary, but early applications of PRM use uniform random sampling. Advanced sampling schemes, such as Gaussian distribution sampling or grid-based sampling, have advantages in certain environments. Determining the number of nodes to sample and an efficient sampling scheme aids in successful construction of the graph, but is often one of the most challenging aspects. While the number of nodes is often a user-selected parameter,

incremental sampling and regularly evaluating the graph based on a metric such as the coverage or density of nodes can help select the ideal number of samples autonomously.[11–13]

After nodes are sampled and added to the graph, connections are then created between nodes by a local planner to form the local paths, or edges, of the graph. Edges can either be undirected, where the nodes in the edge pair can be traversed in either direction, or directed, where the nodes can only be traversed in one direction. These edges are also checked to ensure all local paths adhere to the dynamic and static constraints. Local planners create edges by identifying neighboring nodes to connect. Neighbors may be identified using a variety of methods, such as k-nearest neighbors or radius-based neighbors that depend on a specified distance metric. Edges may also be weighted to reflect desired characteristics of the environment and represent the likelihood or difficulty of traversing the specified edge. In challenging environments, an additional step of refining the roadmap may be included to expand the graph into sensitive or under-represented areas of the solution space, thereby improving the likelihood of success for multiple path planning queries. After these areas are identified, additional nodes and edges are constructed to increase the graph coverage and connectivity in those areas to represent the environment better. After enough nodes and edges are added to the graph, such that the solution space is well covered by nodes and well connected by edges, the roadmap is considered completed. Given a completed roadmap, the query phase searches for a path, a combination of nodes and edges, across the graph from a user-specified start configuration to a goal configuration using a graph search method.

A conceptual example of path planning using PRM appears in Figure 1. Using PRM and a graph search algorithm, the goal in this example is to find a path between the start node, shown as the blue circle, and the goal node, shown as the red circle. First, nodes, shown as grey circles, are sampled from the environment. Only valid nodes, those that do not intersect with any known obstacles which are shown as black squares, are kept in the graph. Then, nodes are connected to nearby nodes to construct directed edges. Each edge is also assigned an edge weight, representing the ease or challenge of traversing a specific edge. Finally, the completed graph is searched for a valid solution path that minimizes the sum of the edge weights from the start to goal.

Dijkstra's Search Algorithm

A suitable graph search method must be selected according to the goal of a specific path planning problem. Classical discrete search algorithms are categorized by the direction the graph is searched, typically either by expanding uniformly outward from the origin or exploiting one direction. In this

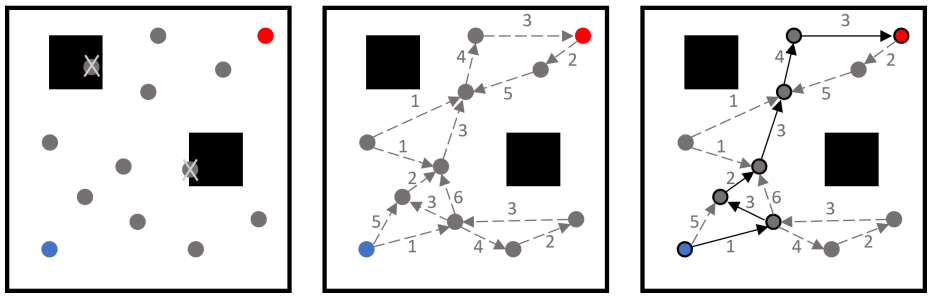

Figure 1. Example of the probabilistic roadmap planning algorithm: a) sample valid nodes, b) connect nodes to create directed edges and assign edge weights, and c) search the completed roadmap for an optimal solution path.

paper, Dijkstra's algorithm, an uninformed best first search algorithm, is utilized for its adaptability of the cost function that determines how the graph is searched. Dijkstra's algorithm is a computationally efficient and complete graph search method. It is proven to be computationally efficient because the algorithm does not need to visit every node in the graph to guarantee it will find the optimal path with respect to a given heuristic.[2] If no solution exists within the graph, the algorithm will terminate in finite time, which makes it a complete algorithm.

Dijkstra's algorithm searches through the graph efficiently by sorting nodes and edges based on the known total path cost, where promising nodes and edges have a lower cost associated with them. The algorithm creates a search tree of the roadmap and two lists are created: a priority list and a closed list. Beginning with the user-defined start node, every neighboring node (determined by the directed edges away from the initial node) is added to the priority list. These nodes and edges are then sorted within the priority list based on the total cost from the start node to the current node. Then, the most promising node is expanded to find its neighbors, which are added to the priority list and again sorted. Nodes that have already been searched and expanded are added to the closed list. This process is repeated until the user-defined goal node has been reached or until all nodes have been searched and are contained within the closed list. The iterative search process for Dijkstra's algorithm involves the following steps:

1) Add the start node(s) to the priority list and sort based on initial node cost, if applicable
2) Remove the best start node from the priority list, add it to the closed list, and add all of its neighboring nodes to the priority list
3) Sort the priority list by the total path cost for each node in the priority list
4) Remove the best node with the lowest total cost and add all neighboring nodes that do not already exist in the priority list
5) Sort and repeat until the goal node(s) is reached or all nodes have been searched and are contained within the closed list

TECHNICAL APPROACH

The application of PRM to transfer design in the Earth-Moon CR3BP in this paper follows a similar general structure to other applications: including both a learning phase where a roadmap is constructed and a query phase where the roadmap is searched to construct an initial guess for a path prior to smoothing. However, this paper also uses a pre-processing phase where stable and unstable manifolds of selected periodic orbits are generated. Using this information, the roadmap is constructed using both states along these known dynamical structures that govern natural transport and states across the broader phase space near the Moon. This approach provides some biasing in the sampling scheme to strategically capture motion along natural transport mechanisms. Then, in the query phase, Dijkstra's algorithm is used to search the roadmap for initial guesses for transfers between selected periodic orbits. Once the initial guesses for transfers are constructed, a multiple shooting algorithm is used to recover continuous natural and maneuver-enabled transfers. A conceptual overview of this process appears in Figure 2. The specific details of these three main phases are presented in the remainder of this section.

Phase 1: Preprocessing Phase

The first phase in this application of using PRM for trajectory design is labeled a preprocessing phase. Within this phase, natural dynamical structures, such as periodic orbits and segments of their associated stable or unstable invariant manifolds are computed at selected values of the Jacobi

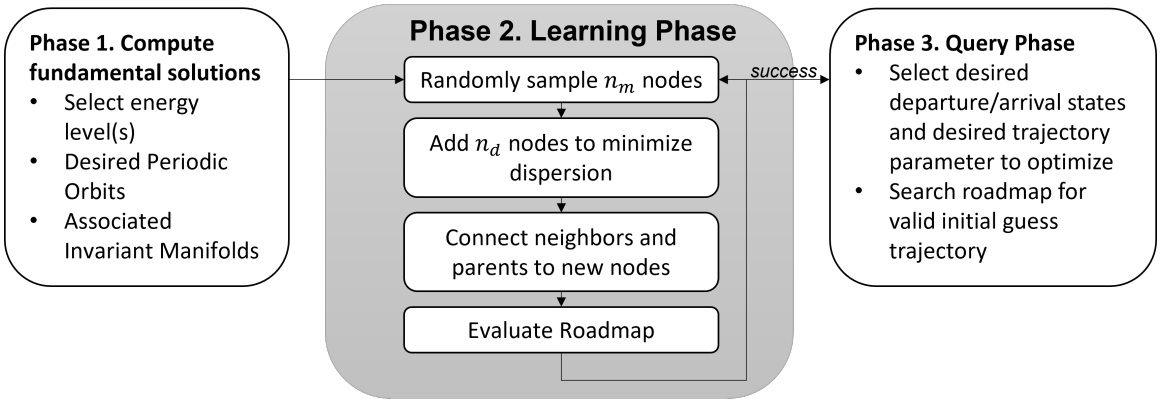

Figure 2. An overview of the three main phases in roadmap generation used to construct initial guesses for transfers between two periodic orbits in the CR3BP.

constant. These fundamental solutions are used to infuse information about natural transport mechanisms into the roadmap. First, selected periodic orbits are discretized evenly in time. These states along the periodic orbits are typically also used to select the start and goal configurations within the roadmap for transfers between periodic orbits. Next, trajectories along the segments of the associated desired invariant manifolds are computed: each state along the periodic orbit is perturbed in the direction of the stable or unstable eigenvector of the monodromy matrix and propagated backward or forward in time. These trajectories are generated until either a selected number of positive crossings of the $x = 1 - \mu$ hyperplane or passage through the L_1 or L_2 gateway. Finally, to support constructing an effective roadmap that includes a sufficiently diverse array of states and arcs along these manifolds, any state along a stable or unstable manifold is described by both its state vector in the rotating frame and its cumulative arclength from the initial condition.

Phase 2: Learning Phase

The second phase is the learning phase, where the roadmap is constructed to approximate the solution space. In this application of transfer design in the Earth-Moon CR3BP, the roadmap is a graph consisting of nodes, defined as spacecraft states expressed in the Earth-Moon rotating frame, and edges, defined as valid natural or maneuver-enabled arcs between selected neighboring nodes. Each edge also has an associated edge weight, defined to reflect desired transfer characteristics. To determine a sufficient size of the roadmap without a priori knowledge of the environment, nodes and edges are added incrementally. After each increment, a roadmap evaluation method is used to assess the coverage and connectivity of the roadmap. This approach is explained in more detail within this subsection.

To initialize the roadmap, nodes are sampled from the trajectories that lie along the selected stable and unstable manifolds computed in the pre-processing phase. These nodes are selected via random sampling; an approach that is straightforward and sufficient for initialization without additional parameters needing to be specified for a more advanced sampling scheme, such as selecting a grid size for grid-based sampling. To implement this random sampling, a node is generated by randomly selecting a trajectory from the entire set of precomputed trajectories. Then, the state that lies along that trajectory at a randomly selected value of the cumulative arc length is selected as a potential node. Before the potential node can be added to the graph, the state must be checked for any possible constraint violations. For this paper, the only node constraint is that its altitude above the

Moon must be above 500km; however, additional constraints may be implemented as needed. If it does not violate any node constraints, the state is added to the graph as a node. To promote the inclusion of natural spacecraft motion within the graph, each valid node is also propagated for a specified arc length and its final state added as an additional node if it does not violate any node constraints. The arc length, l, satisfies the following expression:

$$l = \int_{t_0}^{t_f} \|v(t)\| dt \tag{7}$$

This natural arc that connects the two nodes is saved as a natural edge with an edge weight of 0 because no impulsive maneuvers are required. Note that computing these natural trajectory arcs with a constant arc length, as opposed to a constant integration time, promotes edge diversity in the roadmap via variation in the time of flight.

After a desired number of valid nodes and natural edges are used to initialize the roadmap, additional nodes are added to minimize the graph's dispersion throughout poorly sampled areas of the broader phase space. Dispersion is defined as the radius of the largest empty (not containing nodes) ball within the graph.[14] These areas are identified using a Delaunay triangulation between the current nodes in the graph and calculated in the 3-dimensional configuration space; exploring the application of this approach to the full 6-dimensional phase space is an avenue of ongoing work.

Delaunay triangulation is a common method used within graph-based path planning methods to aid in the construction of the roadmap by identifying candidate nodes or edges within the solution space.[15,16] Given a set of nodes, v, the Delaunay triangulation creates a set, $DT(v)$, of triangles (in 2-dimensional space) or tetrahedra (in 3-dimensional space) such that no nodes are within the circumcircle of any triangle or tetrahedron in $DT(v)$. Each tetrahedron created maximizes the minimum angle of all the angles within the tetrahedron in the triangulation. Once $DT(v)$ is constructed, the volume of each tetrahedron supplies information about the size of the encompassed region of the configuration space that does not contain a node. The volume contained within each of the tetrahedra is, therefore, used to identify the regions of the configuration space that are under-sampled areas by the roadmap.

For this analysis, the centroid of any tetrahedron with a volume that is greater than the top 5% of the volumes of all tetrahedra produces the position vector of a potential node to add to the graph. Then, the speed is determined for the selected position vector using the Jacobi constant expression (Eq. 2); if the roadmap spans multiple energy levels, a randomly selected value of the Jacobi constant is employed. To select the velocity direction, two angles are randomly sampled to produce a velocity unit vector. Multiplying this unit vector by the speed produces the velocity vector of the potential node. If this state vector satisfies the node constraints, it is added to the graph as a node. By computing the Delaunay triangulation in configuration space, previously unreachable areas are connected to the graph through new random motion by the selection of the velocity vector. This may help the trajectory designer explore additional motion in addition to the natural flow described by the dynamical structures computed during the first phase. A conceptual depiction of this process in two-dimensions appears in Figure 3. For 20 randomly sampled points, indicated by the blues circles, a Delaunay triangulation is computed, indicated by the blue lines. The centroid of each triangle, indicated by the black circles, is computed using the vertices of each triangle. The only triangle with an area that is greater than the largest 5% of all triangles is indicated by the red centroid. This centroid then produces the position vector for a candidate for a new node to add to the graph.

Next, each node is connected to up to *k* forward neighbors and *k* backward neighbors by adding

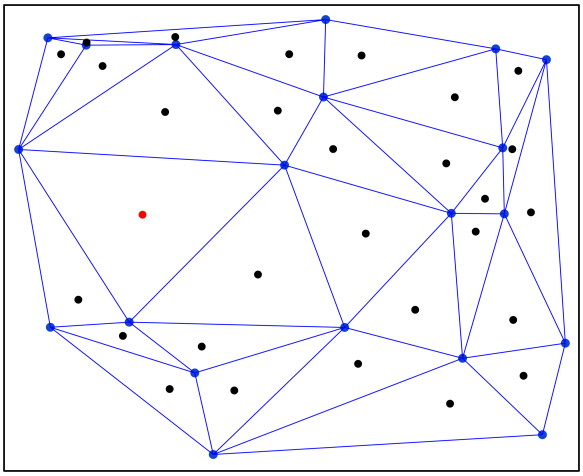

Figure 3. A conceptual example of adding nodes to minimize dispersion using Delaunay triangulation to identify poorly sampled regions of the solution space.

maneuver-enabled edges that flows away from the selected node in forward or backward time, respectively. This process of connecting a node to its neighbors in forward or backward time via maneuver-enabled edges is conceptually demonstrated in Figure 4 for $k = 2$. In this figure, light gray curves correspond to trajectories along a manifold, black curves indicate existing natural edges, and circles depict the states associated with the current set of nodes within the graph. First, to identify neighboring nodes in forward or backward time, the state associated with a selected node is propagated for the same arc length as the natural edges in forward or backward time, respectively. In Figure 4a), a selected node is indicated by a blue circle and the propagated arc is displayed as a dashed black curve. Because this node is not associated with any natural edges in forward time, it is connected to its k nearest neighbors in forward time. These neighboring nodes are identified in configuration space relative to the final state along this dashed black arc, indicated by a small black circle. In Figure 4a), the distances between the final state along this arc and the states of the two nearest nodes are depicted using light gray dotted lines. The identified neighbors, highlighted as green circles in Figure 4a), are then connected to the selected node using maneuver-enabled edges that are calculated via single shooting: the connecting arc must pass through the position vectors of each node, but the velocities at each node and the integration time along the arc may vary. The velocity differences between the states at the selected nodes and the beginning and end of the transfer correspond to two impulsive maneuvers. If this single shooting approach produces a connecting arc, the fmincon tool within MATLAB® is used to minimize the sum of the squares of the maneuver magnitudes. Each connecting arc is also subject to an altitude constraint around the Moon; if a periapsis event exists along the arc and it falls within the desired altitude, then the edge is not connected. If a connecting arc is not recovered either by a failure to converge within the single shooting approach or a constraint violation, then a new neighbor will be identified during the next iteration of the learning phase. These computed maneuver-enabled arcs are displayed as green curves in Figure 4a). Then, the sum of the maneuver magnitudes for the edge supplies the edge weight. This process is repeated until each node in the graph is connected to up to k neighbors in forward time, as depicted in Figure 4b). Next, the process is repeated to produce edges associated with maneuver-enabled arcs that are generated in backward time. In this case, these arcs are displayed in pink in Figure 4c). Following the addition of up to k edges to each node in backward time, the collection of

edges within the current roadmap is depicted in Figure 4d).

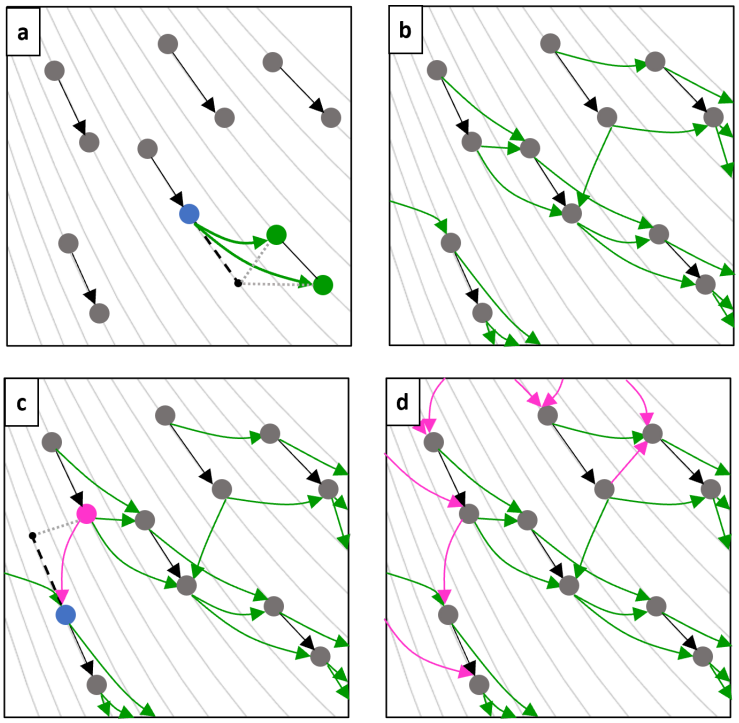

Figure 4. For $k = 2$ after nodes and natural edges are constructed: a)Neighbors forward in time (green) are identified for a node (blue) and connected through maneuver-enabled edges b) All nodes gain up to k neighbors forwards in time, c) One additional neighbor backwards in time (pink) is identified for a node (blue) without k neighbors backwards in time, and d) All nodes gain up to k neighbors backwards in time.

After the desired number of nodes and edges are constructed, the roadmap is evaluated similar to the coverage evaluation and connectivity evaluation metrics successfully demonstrated by Morales, Thomas, and Amato.[13] Coverage aims to ensure that every newly sampled state can be directly connected to the graph and connectivity aims to ensure a path can be found between any desired start and goal configurations within the solution space.[13] By maximizing the coverage and connectivity of the graph, the roadmap is constructed until the graph that sufficiently summarizes the solution space. First, a specified number of randomly-sampled test nodes, subject to the same constraints as the roadmap, are generated within the phase space. For each test node, k forward neighbors and k backward neighbors are identified, using the approach outlined in the previous step. Arcs connecting each test node to each neighbor are then computed and used to assess how well the associated region of the phase space is represented by the roadmap in both forward and backward in time. For neighbors identified forward and backwards in time, edges are only considered successful if the maneuver magnitude associated with the new connecting edge is less than the maximum of the new neighbor's existing edges within the graph. If more than a desired percentage of test nodes are successfully connected to their k forward and k backward neighbors, then the roadmap is considered sufficiently connected with sufficient coverage of the phase space. If not, the iterative process of adding nodes continues. At each iteration, two sets of nodes are added: 1) a selected number of nodes are randomly selected from the stable and unstable manifolds and 2) additional nodes are added to minimize dispersion throughout the undersampled regions of the broader configuration

space using Delaunay triangulation. Using this roadmap evaluation method eliminates the need for a trajectory designer to select how large the graph should be in order to sufficiently cover the solution space; a common challenge across various applications of path planning problems.[1]

Phase 3: Query Phase

After the roadmap is generated, the graph is searched using Dijkstra's algorithm to produce a path between two selected nodes, if one exists. This path through the graph corresponds to a sequence of nodes that connect the start configuration(s) to the goal configuration(s) via a sequences of edges that possess the minimum total edge weight. To design transfers between two periodic orbits in the Earth-Moon CR3BP, these start and goal configurations are selected as one or more nodes associated with states that lie along the periodic orbits. This path within the graph then seeds the initial guess for a transfer that is smoothed using multiple-shooting: the nodes produce states, the edges define the integration time associated with each state, and the user must specify any desired maneuver locations. In this paper, one initial guess is produced from a single search query that is generated through the roadmap. However, ongoing work includes generating multiple viable paths through the roadmap that may connect two configurations.

RESULTS

Initial guesses for transfers between L_1 and L_2 periodic orbits are constructed using PRM in the Earth-Moon CR3BP. To demonstrate the approach in this paper, an initial guess for a transfer from an L_1 Lyapunov orbit to an L_2 Lyapunov orbit is constructed from a roadmap generated at $C_J = 3.15$ for planar motion. The initial guess is then corrected via a multiple shooting corrections method to recover a natural, continuous transfer. At this energy level, two direct heteroclinic connections exist, thereby supporting straightforward verification of the results.[17] Next, a roadmap is generated to construct an initial guess for a spatial transfer between an L_1 northern halo orbit and an L_2 northern halo orbit at $C_J = 3.15$. Because a direct heteroclinic connection does not exist between these two halo orbits, a maneuver-enabled trajectory is recovered.[18] The last scenario focuses on generating a roadmap to construct transfers between L_1 and L_2 halo orbits across a range of energy levels; this roadmap incorporates information about their associated unstable and stable invariant manifolds. This larger, extended roadmap is then searched for initial guesses for transfers at various energy levels between varying halo orbits to recover multiple maneuver-enabled transfers via multiple shooting.

Parameters Governing Roadmap Construction

The presented approach to applying PRM to transfer design in the CR3BP requires selection of parameters that govern the construction of the roadmap. The arc length dictates the length of the natural edges constructed in the graph and is also used to identify neighbors during edge construction. For this paper, an arc length of 0.01 is selected. By using a fixed arc length for manifold sampling and edge construction, the edges exhibit diversity in the integration time and the nodes are better distributed across regions of distinct sensitivities. During the learning phase, a fixed number of nodes are sampled from the precomputed fundamental structures. For iteration 1, states are sampled until 250 valid nodes are added to the graph, while only 100 nodes are added in subsequent iterations. By sampling more nodes upfront, more meaningful edges are constructed given a denser graph. When nodes are added to the graph to minimize dispersion, the largest 5% tetrahedra

are identified as the outliers. This resulted in a sufficient number of new nodes in poorly sampled regions, without adding too many new nodes that may lie close together in neighboring tetrahedra.

Next, recall that the number of edges per node both forwards and backwards in time is denoted as k for this analysis. Manual iteration on the value of k reveals that selecting a low number for k results in too few desirable connections between nodes and poor initial guess construction, while a higher number for k resulted in redundant, high-cost connections and a large increase in computational time. For the examples explored in this paper, a value of $k = 3$ was found to produce an adequate amount of connectivity within the roadmap. The last set of parameters required to select are the number of test nodes sampled during the roadmap evaluation as well as the success criteria. For each evaluation, 50 states are sampled as test nodes and connected to $k = 3$ neighbors forwards and backwards in time, resulting in 150 edges from neighbors identified both forward and backward in time. In order for the roadmap to be considered completed, 75% of edges connected to the neighbors forward in time and 75% of edges connected to the neighbors backward in time must be successful. These roadmap evaluation parameters were selected to maximize the coverage and connectivity of the graph, while reducing the amount of computational complexity. These parameters are constant across all scenarios examined in this paper within the Earth-Moon CR3BP. While these parameters are sufficient for the examples explored within this paper, the exploration of autonomously selecting these parameters for a given solution space is ongoing work.

Planar Transfers Between an L_1 Lyapunov Orbit and an L_2 Lyapunov Orbit at $C_J = 3.15$

To bias the node sampling scheme when constructing a roadmap that will produce a transfer from an L_1 Lyapunov orbit to an L_2 Lyapunov orbit at $C_J = 3.15$, selected natural transport mechanisms are generated. Each of these two periodic orbits is computed via a multiple shooting corrections method and discretized evenly in time into 50 states along the orbit. These states along each orbit supply the set of start nodes and final nodes used during the graph search. Then, 50 trajectories along the unstable manifold associated with the L_1 Lyapunov orbit and the stable manifold associated with the L_2 Lyapunov orbit are generated. Each trajectory is integrated for either 2 positive crossings of the $x = 1 - \mu$ hyperplane, if the trajectory passes through the L_1 or L_2 gateway, or if the trajectory intersects the Moon.

To initialize the roadmap during the learning phase, 250 nodes are generated by randomly sampling the manifold data. Each of these nodes gains a natural edge and an additional neighboring node at the end of the natural edge with an arc length of 0.01 nondimensional units, so the graph is initialized with 500 nodes and 250 natural edges. Initialization requires a larger number of initial nodes to aid in the connections of neighbors and parents. If too few nodes are initially added to the graph, unrealistic edges might be created during the first iteration. Then, additional nodes are added to minimize dispersion throughout poorly sampled regions of the configuration space. The position vectors of these nodes are identified using the centroids of the tetrahedra calculated through Delaunay triangulation in the configuration space that possess volumes in the 95th percentile; their velocity directions are randomly generated. After additional nodes are added in poorly sampled areas of the graph, each node gains up to $k = 3$ forward neighbors and $k = 3$ backward neighbors that are connected with maneuver-enabled edges. Last, the roadmap evaluation method is applied using 50 test nodes. If 75% of the possible connections made in each forwards and backwards time are successful, then the roadmap is considered completed. If not enough test edges are successful, the learning phase iterates again, adding 100 nodes sampled from the manifold data, resulting in 200 new nodes and 100 new natural edges, and adding a variable number of nodes to minimize dis-

persion throughout the broader phase space. While constructing the roadmap in an iterative manner may require a longer or unspecified computational time, it aids the trajectory designer by reducing the a priori knowledge required to effectively construct a sufficient graph spanning regions of distinct sensitivity. For this example, after 2 iterations, the graph is completed with 1887 nodes and 6746 edges constructed. The completed graph's distribution of nodes in the configuration space after the first iteration and after the final iteration are depicted in Figure 5a) and b), respectively. The completed roadmap, including both nodes and edges, appears in Figure 6. Nodes and edges that were constructed outside the selected orbits, such as the cluster of nodes sampled above the L_2 orbit, were sampled along trajectories that pass through the L_1 or L_2 gateway. In all three figures, nodes are indicated by a blue circle, while edges are indicated by a blue curve.

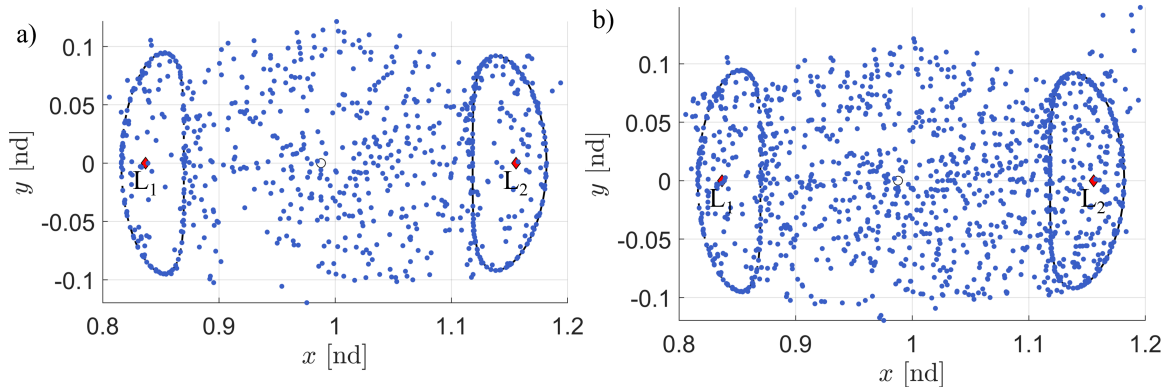

Figure 5. a) Distribution of nodes after the first iteration of the learning phase b) Distribution of nodes after the roadmap is completed.

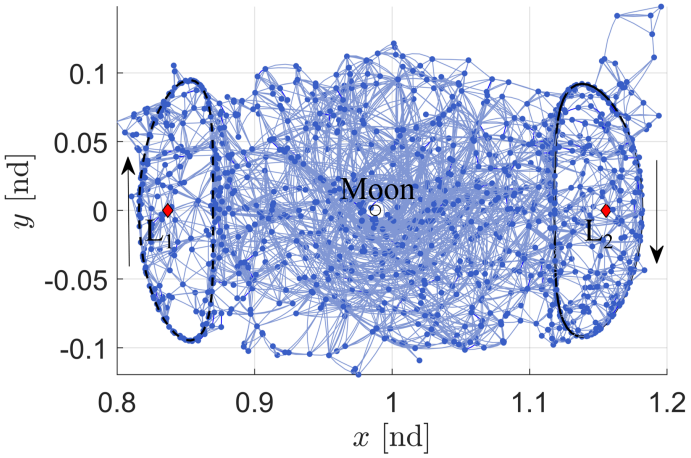

Figure 6. A completed roadmap generated from planar Lyapunov orbits and their associated manifolds at $C_J = 3.15$.

Given the completed roadmap, the graph is searched using Dijkstra's algorithm for the minimum total maneuver magnitude transfer departing anywhere along the L_1 Lyapunov orbit and arriving anywhere along the L_2 Lyapunov orbit. This search produces a sequence of nodes between any two states along the initial and final orbits that are connected by edges that possess the minimal cumulative edge weight. The states at each node and the integration time form the initial guess

for the transfer whereas the edge weight informs manual selection of allowable maneuver locations. The discontinuous initial guess, constructed from the completed roadmap in Figure 6, is displayed in Figure 7a) in the configuration space. The initial and final Lyapunov orbits are plotted with dashed and solid black lines, respectively. The states at each node are indicated by a blue circle while the arcs associated with each edge appear as blue curves. The initial and final state along the initial guess are highlighted with red asterisks. The associated edge weight costs are summarized in Table 1. Given that these edge weights produce a reasonably low total maneuver magnitude, this initial guess is used to recover a natural transfer. In a general scenario, there is no guarantee that a certain cumulative edge weight will indicate the existence of a natural transfer. However, in this scenario, two heteroclinic connections exist at this energy level. If an initial guess does not exist sufficiently close to a natural transfer, then the corrections approach will likely fail. In that case, a maneuver-enabled transfer may be sought and corrections repeated. In addition, continuation or optimization may be employed to determine the lowest total maneuver magnitude. In any case, the initial guess generated in this example is successfully corrected using a multiple shooting corrections method to produce the natural transfer that appears in blue in Figure 7b) and closely resembles a known heteroclinic connection; the initial guess is overlayed in gray for reference. The geometry of the initial guess closely follows the corrected natural transfer, demonstrating the success of constructing a good initial guess for a transfer between Lyapunov orbits in the Earth-Moon CR3BP via the presented technical approach.

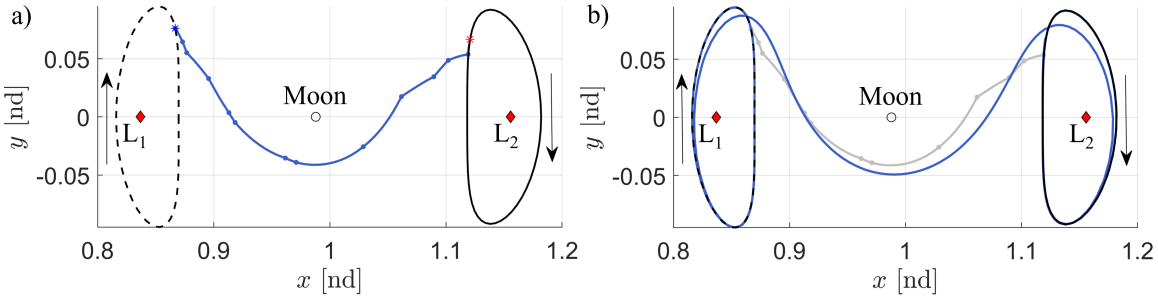

Figure 7. a) The initial guess constructed from the roadmap in Figure 6 and b) the corrected continuous natural transfer at $C_J = 3.15$

Table 1. Edge weight cost in m/s for initial guess shown in Figure 7

Arc 1	3.16	Arc 4	3.61	Arc 7	0	Arc 10	12.20	Arc 13	0.32
Arc 2	0	Arc 5	0	Arc 8	2.62	Arc 11	4.86	**Total Cost**	
Arc 3	10.77	Arc 6	7.67	Arc 9	5.74	Arc 12	10.20	61.14 m/s	

Transfers Between an L_1 Northern Halo Orbit and an L_2 Northern Halo Orbit at $C_J = 3.15$

A new roadmap is generated to construct an initial guess for a maneuver-enabled transfer in a more complex example: for spatial motion from an L_1 northern halo orbit to an L_2 northern halo orbit at $C_J = 3.15$. Each orbit is again discretized into 50 states evenly distributed in time and each state is added as a node to the graph. Then, 50 trajectories along the unstable manifold associated with the L_1 halo orbit and 50 trajectories along the stable manifold associated with the L_2 halo orbit are computed to bias node sampling. The roadmap is constructed during the iterative learning

phase, using the same roadmap construction parameters as the previous example. Following this process, after 2 iterations, the graph is considered completed with 2283 nodes and 8046 edges. The completed graph's distribution of nodes after the first iteration and after the final iteration are shown separately in Figure 8a) and b), respectively, while the completed roadmap appears in Figure 9; the node and edge markers and color scheme are consistent with Figure 6. While the roadmap covers the solution space governed by the manifolds structures used to bias the sampling, motion outside of these structures is not as well-summarized. Since this motion could provide additional maneuver-enabled transfers that do not lie within the manifold structure, additional graph construction will be required in those regions, which will be explored in future work.

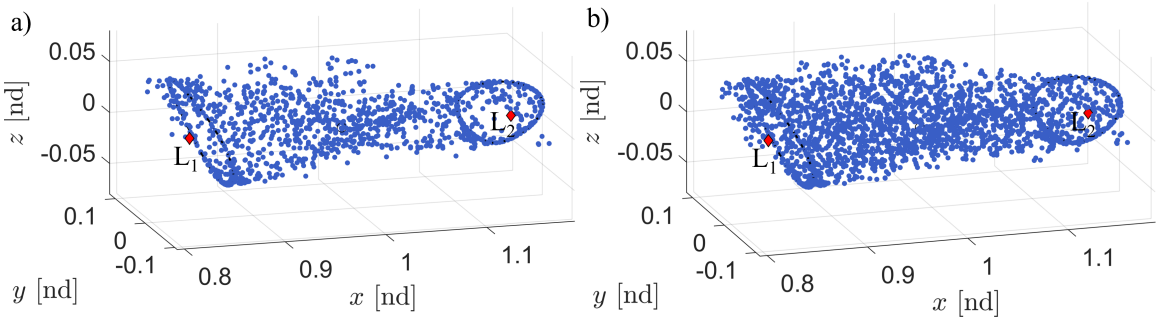

Figure 8. Halo transfer roadmap: a) Distribution of nodes after the first iteration of the learning phase b) Distribution of nodes after the roadmap is completed.

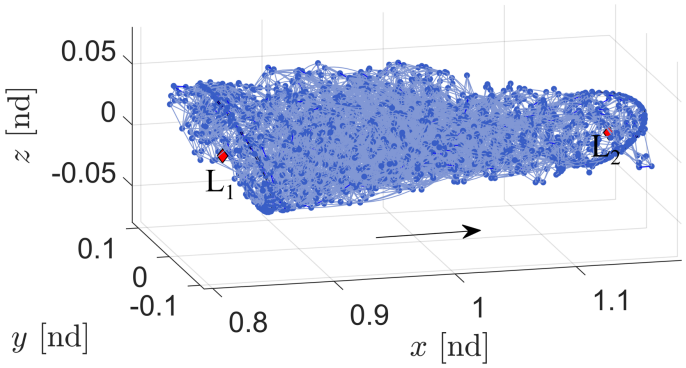

Figure 9. A completed roadmap generated from northern halo orbits and their associated manifolds at $C_J = 3.15$.

Once the roadmap is generated, an initial guess is constructed by searching the graph for the path with the lowest cumulative edge weight. This initial guess appears in Figure 10a) using a configuration that is consistent with Figure 7. The associated edge weights are reported in Table 2. Using Table 2 as a reference, arcs 6-7 near the Moon and arcs 10-12 that occur prior to arrival into the L_2 halo orbit appear to have higher edge weights than other arcs. Based on this insight, consider allowing 1 maneuver near the Moon, between the 7th and 8th arc, near the $x = 1 - \mu$ hyperplane. In this example, the transfer is constrained to depart naturally from the initial L_1 halo orbit and arrive naturally onto the final L_2 halo orbit. Using these states, integration times and single maneuver location, the initial guess is corrected using a multiple shooting corrections algorithm to produce the transfer depicted in 10b). The initial guess deviates slightly from the corrected transfer, which is

expected given that the initial guess would require multiple smaller maneuvers whereas the correct transfer uses one maneuver, indicated by the red circle with a magnitude of $14.18m/s$. However, their general geometries are similar, indicating the value of a PRM approach to constructing useful initial guesses for transfers in this scenario.

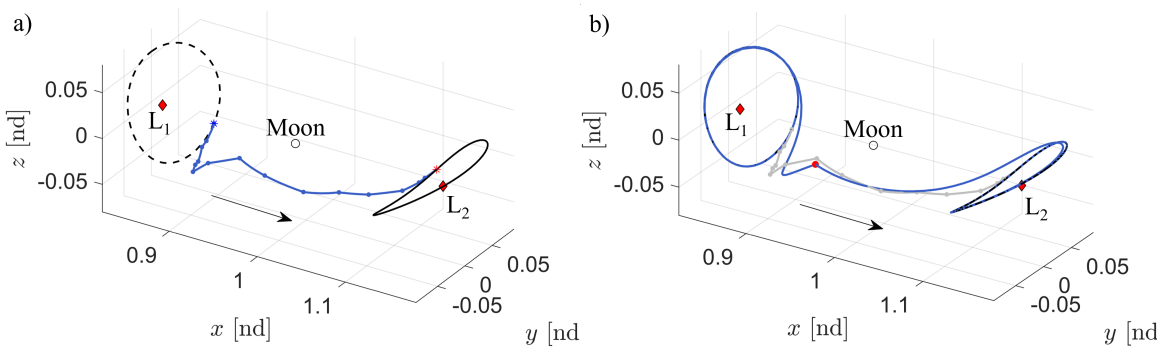

Figure 10. a) The initial guess constructed from the roadmap in Figure 9 and b) the corrected continuous transfer at $C_J = 3.15$ with one maneuver.

Table 2. Edge weight cost in m/s for initial guess shown in Figure 10

Arc 1	2.56	Arc 4	0	Arc 7	14.98	Arc 10	12.69	Arc 13	3.62
Arc 2	0	Arc 5	4.46	Arc 8	7.83	Arc 11	11.20	Arc 14	0
Arc 3	8.46	Arc 6	17.69	Arc 9	8.18	Arc 12	15.03	Arc 15	1.65
Total Cost				118.37 m/s					

Spatial Transfers Across Multiple Energy Levels

A main benefit of PRM over other sampling-based algorithms is the capability to construct a roadmap once and search the graph for multiple queries to produce unique solutions. Using the graph in this manner allows a trajectory designer to rapidly explore the solution space for a variety of arrival and departure states. In this example, a larger roadmap is constructed for direct transfers between halo orbits across multiple energy levels in the Earth-Moon CR3BP. In the preprocessing phase, northern halo orbits and trajectories along both the associated unstable and stable invariant manifolds for each orbit are generated for Jacobi constants in the range of $C_J = [3.12, 3.15]$. Similar to the previous examples, 50 states along each of the 8 orbits are computed and 50 trajectories along each associated manifold are computed, totaling 800 pre-computed trajectories.

While the roadmap is constructed using the same approach as the previous example, the velocity vectors of each node added to minimize dispersion are now elected to produce a randomly generated Jacobi constant within the range $C_J = [3.12, 3.15]$. The roadmap generation procedure applied to this example requires 5 iterations to produce a completed graph with 4038 nodes and 15751 edges constructed. It is important to note the increase in the graph size when sampling from a larger subset of information. While additional structures were precomputed, the graph is less than 2.5x larger than the spatial roadmap constructed at one energy level. An example of this extended roadmap is shown in Figure 11 using the same node and edge markers and color scheme as the previous examples. This roadmap is considered complete using the evaluation metric previously

described and is used to successfully construct multiple initial guesses for transfers between orbits. However, a few regions within the solution space appear to be lacking nodes and edges while some regions appear to have redundant nodes and edges. This is expected with the implementation since a large number of trajectories used to bias the sampling are computed within the desired solution space. By using a more efficient representation of the natural dynamics in combination with more randomly generated motion, a more efficient summarized graph of the broader solution space may be constructed. Using randomly generated motion may also aid in the exploration of unknown areas, where dynamical structures may be more complex or impractical. These additional methods to aid in the construction of a more efficient, completed roadmap are being explored in ongoing work.

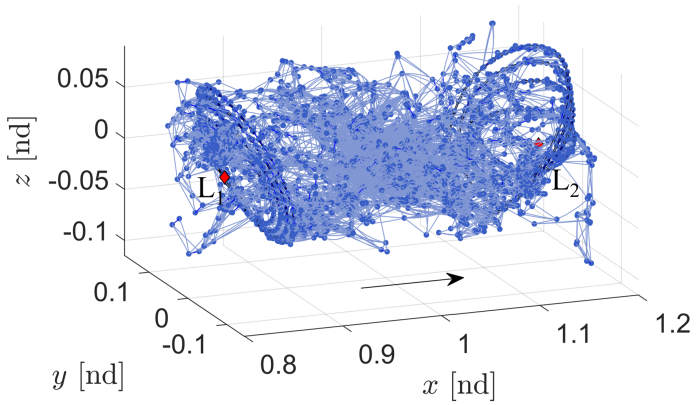

Figure 11. A completed roadmap generated to design transfers between northern halo orbits at $C_J = [3.12 - 3.15]$.

To construct a variety of initial guesses for transfers between the northern halo orbits used to guide construction of the roadmap, multiple search queries are used with varying parameters. The initial orbit and final orbit can vary depending on what energy levels are desired for the transfer, and the initial guess for a specific transfer is allowed to depart and arrive anywhere along the desired orbits. Two examples of initial guesses for transfers from distinct combinations of an L_1 halo orbit to an L_2 halo orbit are shown in Figure 12a) and c); the associated corrected maneuver-enabled transfers appear in Figure 12b) and d). The associated edge weights for each initial guess are reported in Table 3. For the initial guess in Figure 12c), arcs 15-60 all have an edge weight (total maneuver magnitude) of 0 because these edges traverse a neighboring L_2 halo orbit. The corrected transfer in Figure 12b) has three maneuvers: a 52.87 m/s maneuver near the L_1 halo orbit, a 50.34 m/s maneuver near the Moon, and a 61.59 m/s maneuver near the L_2 orbit. Similarly for the corrected transfer in Figure 12d), a 51.58 m/s maneuver is placed near the L_1 halo orbit, a 46.23 m/s maneuver is placed near the Moon, and a 66.17 m/s maneuver is placed near the L_2 orbit. These maneuver locations are manually selected following analysis of the edge weights along the paths generated via the graph search. Analysis of Figure 12 reveals that each corrected transfer remains close in geometry to its initial guess, even though a minimal number of maneuvers were allowed along the final corrected transfer. As another example, an initial guess for a transfer from an L_2 halo orbit to an L_1 halo orbit is constructed and displayed in Figure 13a). The geometry of the initial guess from L_2 to L_1 orbits does not resemble the natural motion along the associated invariant manifolds as expected, which could indicate poor sampling in the direction of motion, suggesting required future work to explore this region of the graph. One corresponding transfer is shown in Figure 13b) with three maneuvers: a 69.93 m/s maneuver near the L_1 halo orbit, a 79.54

m/s maneuver near the Moon, and a 52.45 m/s maneuver near the L_2 orbit. Another transfer with maneuvers allowed between all neighboring arcs along the transfer is shown in Figure 13c), with a total maneuver magnitude of 658.18 m/s. The transfer in Figure 13b) possesses a distinct geometry compared with the initial guess due to the large edge weights, reported in Table 4, of this path, indicating that it is a poor initial guess for the selected number and location of maneuvers. This observation is confirmed by the close resemblance between the transfer in Figure 13c) that uses several maneuvers and the initial guess. Each initial guess can be corrected to recover transfers that are similar in geometry, if a suitable number of maneuvers is allowed, demonstrating the feasibility of using the roadmap to construct spatial transfers.

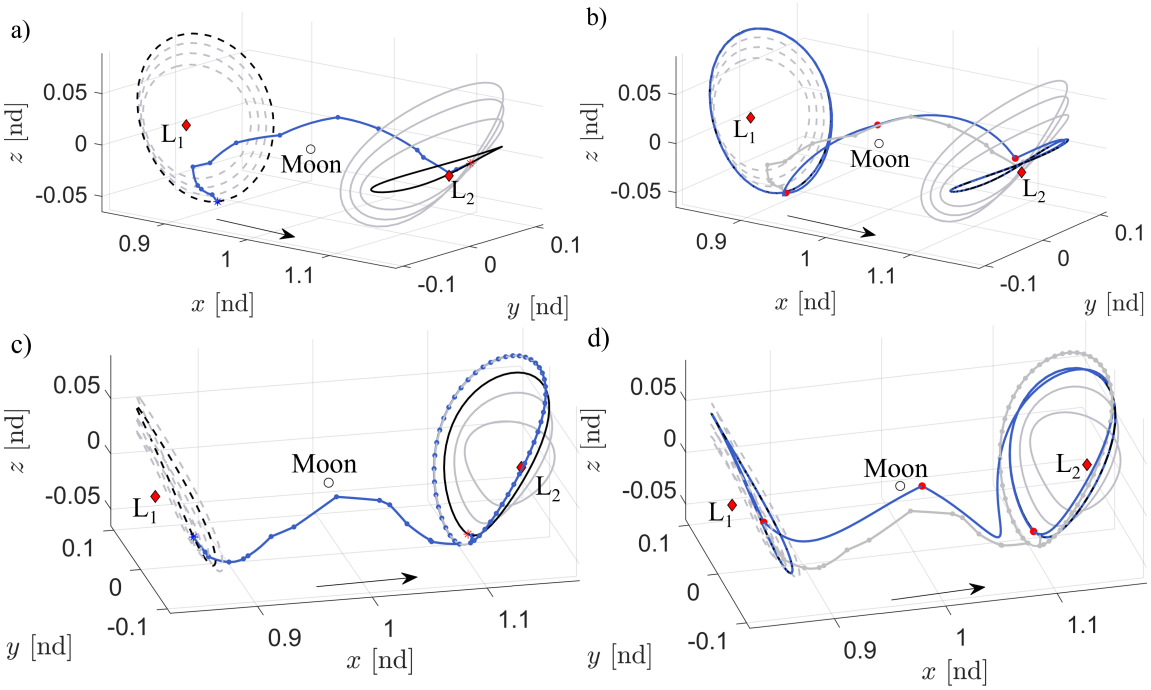

Figure 12. Two initial guess transfers and the corresponding corrected transfer with three maneuvers: a) from an L_1 halo orbit at $C_J = 3.12$ to an L_2 halo orbit at $C_J = 3.15$, b) from an L_1 halo orbit at $C_J = 3.13$ to an L_2 halo orbit at $C_J = 3.13$.

Table 3. Edge weight costs in m/s for initial guesses shown in Figure 12

Arc 1	14.40	Arc 4	12.59	Arc 7	20.26	Arc 10	17.96	Arc 13	0		
Arc 2	2.18	Arc 5	11.19	Arc 8	10.50	Arc 11	0	Arc 14	2.22		
Arc 3	0	Arc 6	9.81	Arc 9	11.52	Arc 12	16.90				
Total Cost for Figure 12a)						129.54 m/s					
Arc 1	2.63	Arc 4	2.18	Arc 7	11.20	Arc 10	0	Arc 13	0	Arc 62	0
Arc 2	0	Arc 5	0	Arc 8	9.81	Arc 11	4.67	Arc 14	26.52	Arc 63	13.36
Arc 3	6.75	Arc 6	12.58	Arc 9	28.81	Arc 12	7.30	Arc 61	0.49	Arc 64	0.33
Total Cost for Figure 12c)						126.64 m/s					

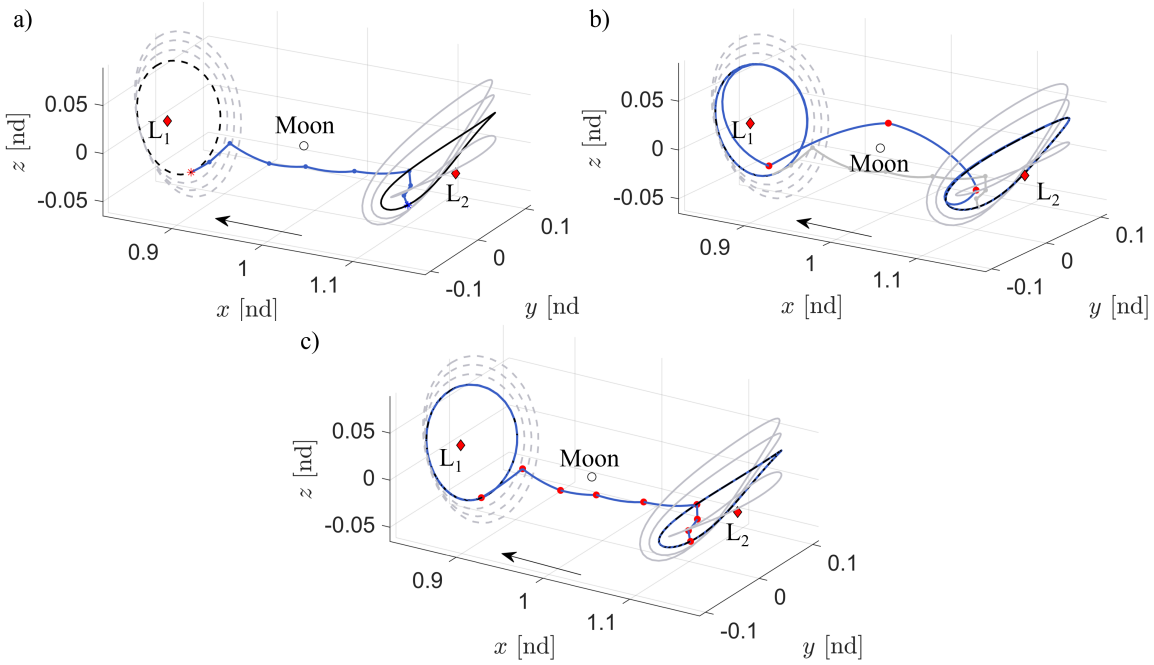

Figure 13. a) An initial guess for a transfer from an L_2 halo orbit at $C_J = 3.15$ to an L_1 halo orbit at $C_J = 3.14$, b) the corresponding corrected transfer with three maneuvers, and c) the corresponding corrected transfer with nine maneuvers.

Table 4. Edge weight cost in m/s for initial guess shown in Figure 13

Arc 1	11.21	Arc 3	31.82	Arc 5	29.74	Arc 7	27.30	Arc 9	13.23
Arc 2	5.58	Arc 4	26.67	Arc 6	14.69	Arc 8	14.67		
Total Cost				174.89 m/s					

CONCLUSION

In this paper, PRM is used to design transfers in the Earth-Moon CR3BP, expanding upon earlier work described in Bruchko and Bosanac.[7] These roadmaps summarize a portion of the solution space near the vicinity of the Moon and support constructing a variety of initial guesses between libration point orbits. PRM methods for trajectory design in the CR3BP involves three phases: the preprocessing phase, the learning phase, and the query phase. During the preprocessing phase, the desired periodic orbits and their associated hyperbolic invariant manifolds are computed. States along these fundamental solutions serve as the data set to partially bias the sampling within the learning phase. Next a roadmap, a weighted and directed graph, is generated in the learning phase. Nodes represent valid spacecraft states sampled partially from the data set of precomputed fundamental solutions as well as within the region of the solution space. Edges represent valid natural or maneuver-enabled trajectory arcs between nodes. Each edge has an associated edge weight that represent the sum of the maneuvers required to traverse from the parent node to the neighboring node. The roadmap is generated through an iterative process of adding nodes and edges. Once the roadmap is completed, the graph is searched using Dijkstra's algorithm to produce a path that connects the desired start node(s) to goal node(s) with the lowest cumulative edge weight. These paths

seed an initial guess for a transfer. This procedure is demonstrated in the context of three examples of increasing complexity: 1) designing natural, planar transfers from an L_1 Lyapunov orbit to an L_2 Lyapunov orbit at $C_J = 3.15$, 2) designing spatial, maneuver-enabled transfers between L_1 and L_2 northern halo orbits at $C_J = 3.15$, and 3) designing spatial, maneuver-enabled transfers between distinct combinations of L_1 and L_2 northern halo orbits across the range $C_J = [3.12, 3.15]$. In each example, the initial guesses were corrected to successfully recover transfers, when a suitable number of maneuvers are allowed. These results motivate the continued exploration of using PRM for trajectory design in more complex solution spaces and different multi-body systems.

ACKNOWLEDGMENT

This research is being completed at the University of Colorado Boulder. It is supported by a NASA Space Technology Graduate Research Opportunity.

REFERENCES

[1] P. Svestka and M. H. Overmars, *Robot Motion Planning and Control*. Berlin: Springer, Berlin, Heidelberg, 1st ed., 1998.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge, UK: Cambridge University Press, 1st ed., 2006.

[3] K. Lynch, G. Kantor, H. Choset, L. Kavraki, S. A. Hutchinson, W. Burgard, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, 1st ed., 2005.

[4] A. Das-Stuart, K. Howell, and D. Folta, "A Rapid Trajectory Design Strategy for Complex Environments Leveraging Attainable Regions and Low-Thrust Capabilities," Adelaide, Australia, 68th International Astronautical Congress, September 2017.

[5] J. A. Starek, E. Schmerling, G. D. Maher, B. W. Barbee, and M. Pavone, "Real-Time, Propellant-Optimized Spacecraft Motion Planning under Clohessy-Whilshire-Hill Dynamics," Big Sky, MT, IEEE Aerospace Conference, March 2016.

[6] E. Trumbauer and B. Villac, "Heuristic Search-Based Framework for Onboard Trajectory Redesign," *Journal of Guidance, Control and Dynamics*, Vol. 37, No. 1, January-February 2014, 10.2514/1.61236.

[7] K. Bruchko and N. Bosanac, "A Preliminary Exploration of Path Planning for Initial Guess Construction in Multi-Body Systems," AAS/AIAA Astrodynamics Specialist Conference, August 2021.

[8] N. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, "OBPRM: An Obstacle Based PRM for 3D workspaces," *Proceedings of the Fourteenth Workshop on the Algorithmic Foundations of Robotics*, Wellesley, MA, 1998.

[9] W. S. Koon, M. W. Lo, J. E. Marsden, and S. D. Ross, *Dynamical Systems, The Three-Body Problem and Space Mission Design*. Marsden Books, 3rd ed., 2006.

[10] N. Bosanac, *Leveraging Natural Dynamical Structures to Explore Multi-Body Systems*. PhD thesis, Purdue University, IN, 2016.

[11] J.-H. Park and T.-W. Yoon, "Maximizing the Coverage of Roadmap Graph for Optimal Motion Planning," *Hindawi*, Vol. 2018, November 2018, 10.1155/2018/9104720.

[12] B. Park and W. K. Chung, "Adaptive Node Sampling Method for Probabilistic Roadmap Planners," St. Louis, USA, IEEE/RSJ International Conference on Intelligent Robots and Systems, October 2009.

[13] S. T. Marco Morales and N. M. Amato, "Incremental Map Generation," Advanced Robotics, January 2006.

[14] L. E. Kavraki and S. M. LaValle, *Motion Planning*. Springer Handbook of Robotics. Springer, Berlin, Heidelberg, 2008, 10.1007/978-3-540-30301-5_6.

[15] Y. Huang and K. Gupta, "A Delaunay Triangulation Based Node Connection Strategy for Probabilistic Roadmap Planners," New Orleans, LA, IEEE International Conference on Robotics and Automation, April 2004.

[16] W. C. T. Gene Eu Jan, Chi-Chia Sun and T.-H. Lin, "An O(nlogn) Shortest Path Algorithm Based on Delaunay Triangulation," *IEEE/ASME Transactions on Mechatronics*, Vol. 19, No. 2, April 2014, 10.1109/TMECH.2013.2252076.

[17] E. Canalias and J. J. Masdemont, "Homoclinic and Heteroclinic Transfer Trajectories between Lyapunov Orbits in the Sun-Earth and Earth-Moon Systems," 10 2007.

[18] A. Haapala, *Trajectory design in the spatial circular restricted three-body problem exploiting higher-dimensional Poincare maps*. PhD thesis, Purdue University, IN, 2014.