



A Low-Complexity Algorithm to Determine Trajectories Within the Circular Restricted Three-Body Problem

David Canales¹ · Sirani M. Perera² · Atahan Kurttisi¹ · Brian Baker-McEville¹

Accepted: 30 September 2023 / Published online: 7 November 2023
© The Author(s), under exclusive licence to American Astronautical Society 2023

Abstract

With the growing volume of traffic within the Cislunar region, there is an increasing need for efficient techniques to propagate trajectories of spacecraft in the circular restricted three-body problem. A low-complexity algorithm utilizing interpolation and incorporating boundary conditions is introduced for generating accurate trajectories and periodic orbits within the Cislunar domain. The proposed approach offers a distinct advantage over existing iterative techniques, as it yields favorable results in terms of arithmetic and time complexities. Once the reliable low-complexity algorithm is developed, it is applied to relevant Cislunar trajectories. Finally, we have compared the time complexity of the proposed algorithm with that of a traditional orbit propagator. The algorithm achieves significant improvements in time complexity for various types of orbital trajectories compared to traditional iterative methods. It demonstrates approximately a 50% enhancement of time efficiency for low-Lunar, Lyapunov, near-rectilinear halo, and distant retrograde orbital trajectories.

Keywords Low-complexity algorithm · Navigation · Cislunar realm · Three-body problem · Periodic orbits

✉ David Canales
canaled4@erau.edu

Sirani M. Perera
pereras@erau.edu

Atahan Kurttisi
kurttisa@my.erau.edu

Brian Baker-McEville
bakermcb@my.erau.edu

¹ Department of Aerospace Engineering, Embry-Riddle Aeronautical University, Daytona Beach 32114, FL, USA

² Department of Mathematics, Embry-Riddle Aeronautical University, Daytona Beach 32114, FL, USA

1 Introduction

The Cislunar region, shown in Fig. 1, is gaining increased attention throughout the past few years, as 90 missions to the Moon are projected by 2030 [1], with additional missions to Mars. Many upcoming Cislunar missions are focused on the Lunar South pole, as well as periodic orbits about L_1 and L_2 of the Earth-Moon circular restricted three-body problem (CR3BP) system [2]. Here, L_1 and L_2 are the equilibrium points of the Earth-Moon system. Russia's Luna 25 (2022) [3], South Korea's KPLO (2022) [4], Japan and India's joint LUPEX (2023) [5], and India's Chandrayaan-3 (2024) [6] all are missions to observe, or land on, polar regions of the Moon. Additionally, NASA has multiple commercial Lunar payload services (CLPS) missions to the Lunar polar region. NASA's Artemis program is a multi-stage program to reestablish a presence on the Moon; the first Artemis mission traveled in a distant retrograde orbit (DRO) about the Moon at the end of 2022 [7].

The dynamics that govern the motion in the CR3BP are highly non-linear and no closed-form solution has yet been derived. To be able to design trajectories in such a model, different numerical methods are required to analyze spacecraft trajectories that satisfy desired behaviors. Many different formulations of differential corrections algorithms exist within the context of targeting schemes [8, 9]. In recent times, with evolving computational techniques, many are based on free variables and constraints for trajectories that need to meet a certain set of conditions. One example consists of a method for generating long baseline solutions using multiple shooting techniques [10]. Therefore, following ongoing work on trajectory generation [11–17], the objective is to find an approximated solution to determine trajectories of the CR3BP with a low-complexity algorithm. To propagate orbits and uncertainty within the context of this non-linear realm, typical numerical integration techniques including Gauss–Legendre, Dormand–Prince, Chebyshev–Picard algorithms [18], Gragg–Bulirsch–Stoer [19], and the Adams–Bashforth method [20] are used. Each

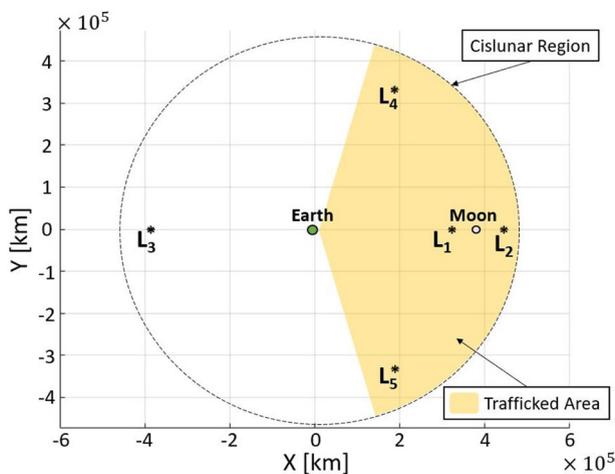


Fig. 1 Cislunar region

technique has its own benefits and disadvantages. Using a higher-order Runge–Kutta integrator (ODE45 in this investigation), the solution is usually more accurate. However, these techniques are typically computationally expensive. It is thus important to reduce the complexity of the problem.

Finite element methods (FEM) and finite difference methods (FDM) are conventional techniques in astrodynamics for solving equations by discretizing spatial data [21, 22]. A trade-off exists between resolution and speed when employing these approaches: coarse discretization provides faster results but sacrifices accuracy, whereas fine discretization improves accuracy at the cost of slower computations. When it comes to solving the equations involved in the CR3BP, traditional solvers face significant challenges due to the requirement of very fine discretization in both time and space. This results in a time-consuming process that can be daunting to handle. We propose a technique that utilizes fine discretization in time and space to determine trajectories in the CR3BP. This offers a more efficient alternative to traditional methods that are more challenging and time-consuming.

Efforts to address the need for efficient propagation methods in the CR3BP have extended beyond typical numerical techniques (see [23–25] and references therein). Researchers provided detailed explanations of the need for low-complexity algorithms [26], and utilized high-dimensional Poincaré maps for spacecraft orbit design in the CR3BP. Existing propagation methods have been compared in [27], including the polynomial approach and the authors designed a differential algebraic method to guarantee access to any point of a family without any numerical propagation. Additional advancements include the utilization of Gaussian process regression for propagation, efficient algorithms for computing invariant manifolds, orbit classification techniques, and nonlinear stability analysis of periodic orbits. In general, these approaches have enabled more precise predictions and analyses in the CR3BP. However, the existing literature and gap indicate the need for further improvements and low-complexity methods dedicated to efficient CR3BP propagation. This paper aims to bridge this gap by proposing a novel technique that incorporates innovative mathematical strategies to enhance efficiency in orbital propagation within the CR3BP framework.

The primary objective of this investigation is thus to develop a low-complexity algorithm that enables efficient trajectory determination within the Cislunar domain, including periodic orbits. This algorithm utilizes predefined boundary conditions instead of iterative methods based on initial conditions. High-fidelity discrete solutions are extracted at finite time steps, constructing a low-complexity propagation between these boundary conditions. The accuracy of the trajectory generated is demonstrated using analytical evidence within the given set of boundary conditions. A comprehensive description of the algorithm derivation and analysis of the arithmetic and time cost savings are provided. To produce analytical solutions and algorithms that are computationally tractable and inexpensive, particularly in the complex and chaotic three-body dynamics, it is important to address system structures in relevant equations, develop novel theories, and design low-complexity and reliable (in the sense of accuracy and stability) algorithms. For obtaining the position, velocity, and acceleration of the spacecraft within the three-body dynamics at defined finite time steps, ODE45 is utilized. The output data from this integration is then identified as

predefined boundary conditions. Subsequently, the low-complexity algorithm in this study is employed to construct the trajectory of the spacecraft within the CR3BP. The study leverages the traditional equations of the CR3BP. Numerical tests are performed for several orbits, including a near-rectilinear halo orbit (NRHO) representing Gateway's orbit, a direct retrograde orbit (DRO) representing part of the Artemis I trajectory, a low-Lunar orbit (LLO), and a Lyapunov orbit about L_2 . The numerical results show that the proposed algorithm achieves approximately 50% improvement in time complexity for all these cases. Therefore, a ground-breaking model for orbit determination in the Cislunar region is achieved.

The paper is structured as follows.¹ First, Sect. 2 presents the dynamics of the circular restricted three-body problem, including supplemental information about differential corrections and the state transition matrix in Appendix A. Then, Sect. 3 develops the theory to obtain a low-complexity and reliable algorithm to describe smooth trajectories of the moving body in the presence of multiple gravitational fields. In Sec. 4, a numerical analysis based on the time complexity of the proposed algorithm is developed. Results are compared with a traditional method of orbit integration (i.e., ODE45) for all suggested orbits. Finally, we conclude the paper in Sect. 5.

2 Dynamical Model

To model the problem, a system of differential equations in the CR3BP is written in dimensionless form [28], in such a way that the characteristic distance is defined as the time-averaged distance between the Earth and the Moon; and the characteristic time is selected to guarantee a dimensionless mean motion of the primaries with a value equal to unity. The mass ratio $\mu = m_M / (m_M + m_E)$ is defined for the system, with m_M and m_E being the masses of the Moon and the Earth, respectively. A barycentric rotating frame is defined with the \hat{x} -axis directed from the Earth-Moon barycenter to the Moon, and the \hat{z} -axis from the barycenter in the direction of the system angular momentum vector. Figure 2 presents a schematic representation of the model. The Earth and Moon are located at positions $\bar{r}_p = [-\mu, 0, 0]^T$ and $\bar{r}_m = [1 - \mu, 0, 0]^T$, respectively. The evolution of a spacecraft (s/c) position $\bar{r}_{rot} = [x, y, z]^T$ and velocity $\dot{\bar{r}}_{rot} = [\dot{x}, \dot{y}, \dot{z}]^T$ is governed by the following equations of motion [29]:

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

where dots indicate derivatives with respect to dimensionless time. Then, $U^* = \frac{1-\mu}{r_{p-s/c}} + \frac{\mu}{r_{m-s/c}} + \frac{1}{2}(x^2 + y^2)$ represents the pseudo-potential function for the system of differential equations; $r_{p-s/c}$ and $r_{m-s/c}$ are the distances of the s/c to the Earth

¹ Part of this investigation was presented in 33rd AAS/AIAA Space Flight Mechanics Meeting, Austin, TX, 2022.

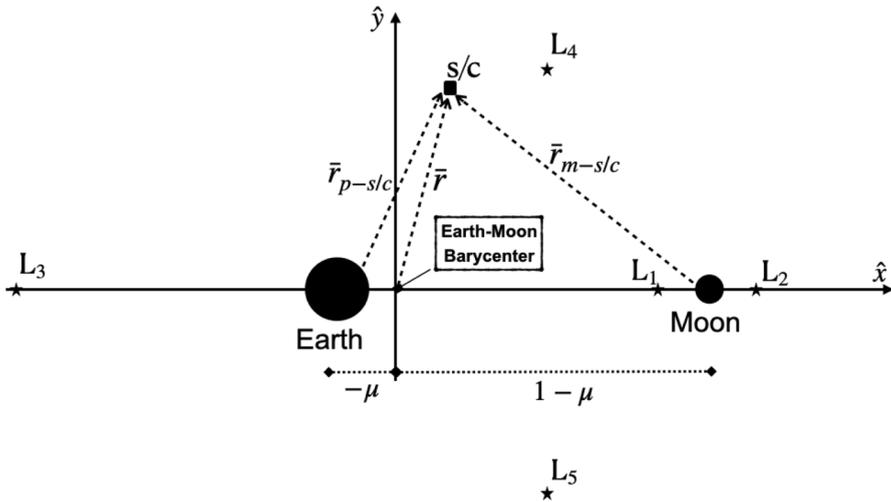


Fig. 2 The CR3BP model represented in the barycentric rotating reference frame with the primaries, the third body and the 5 equilibrium points

and the Moon, respectively. The Jacobi constant (JC) is the only scalar integral for the given system which gives information of the energy of the s/c , i.e., $JC = 2U^* - (\dot{x}^2 + \dot{y}^2 + \dot{z}^2)$. Finally, note that motion exists in the vicinity of five equilibrium solutions in the given formulation. Such equilibrium solutions are denoted as L_1 to L_5 in Fig. 2. Such motion can be categorized by families of periodic orbits [30–32]. These orbits are usually leveraged for many distinct types of mission scenarios. In this research, different periodic orbits are propagated both using a classical integrator as well as the low-complexity algorithm proposed by the authors in the following section. Appendix A provides more information about the targeting scheme used to compute periodic orbits in this research.

3 Low-Complexity Algorithm for Smooth and Continuous Trajectory Determination

Trajectory propagation methods, such as Gauss–Legendre and Dormand–Prince [18], typically generate a trajectory in discrete steps based on a given set of conditions. In contrast, our proposed algorithm generates a trajectory by interpolating a function between a set of boundary conditions. While existing iterative techniques offer accurate solutions, they can be computationally expensive due to the need to compute all predecessors for each different time step. In contrast, the proposed technique generates a trajectory at a given time stamp without the need to compute all previous states from the beginning. This is achieved by using a piecewise continuous function based on polynomial interpolation, which offers a low-complexity solution compared to existing iterative methods. For the dynamics of the spacecraft in the CR3BP, the proposed interpolated polynomial function provides greater accuracy

Table 1 Comparison between interpolation and iterative trajectory propagation methods

Interpolation (proposed)	Iterative (existing)
Solve in continuous domain	Solves in discrete domain
Closed-form accurate solution	Approximated solution
Low-complexity	Computationally expensive

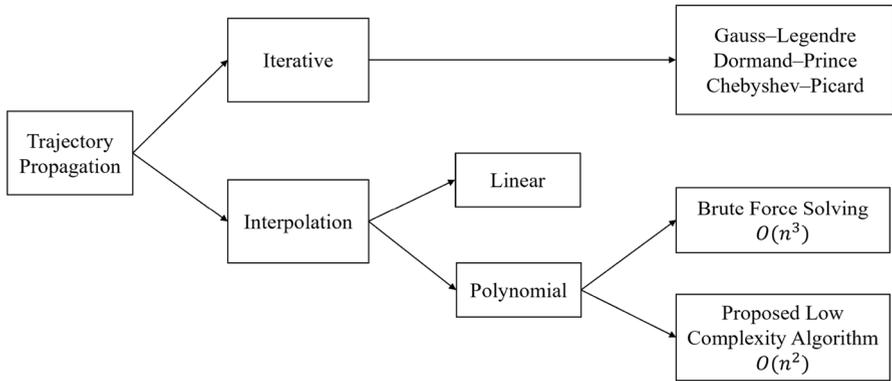


Fig. 3 Trajectory propagation using iterative and interpolation techniques

and computational efficiency than linear functions [33]. The differences between the proposed interpolation technique and iterative methods are represented in Table 1. To achieve computationally efficient and time-effective algorithms, particularly for complex and chaotic three-body dynamics, it is essential to consider system structures that result in low-complexity and reliable (in terms of accuracy and stability) algorithms. In this section, novel piecewise-defined functions are proposed to describe smooth trajectories of the spacecraft in CR3BP. While a linear trajectory is often assumed over a given time interval, in reality, trajectories are polynomials in time due to changes in position, velocity, and acceleration at each time stamp. The proposed algorithm can be extended to fully determine trajectories of the n-body problem at each time step, offering a low-complexity solution, but this investigation is limited to the CR3BP. Figure 3 presents a visual flowchart comparing the different trajectory propagation methods.

In the context of the CR3BP formulation discussed in the previous section, to obtain the position, velocity, and acceleration of the spacecraft within the three-body dynamics at each defined finite time step, ODE45 is employed, and the resulting data is used as predefined boundary conditions. In the CR3BP, the position, velocity, and acceleration data for the spacecraft are known, providing boundary conditions for each time interval. This would be equivalent to having measured states of spacecraft in regards to orbit determination. Using the Lagrange interpolation theorem with the six data points at the interval’s start and end, a unique degree-five polynomial is obtained that best fits or interpolates the data within the interval. Therefore, each body’s trajectory at any given

time interval are uniquely determined as a degree-five polynomial in time, rather than linear in time. The trajectories of the spacecraft in CR3BP are described as piecewise-defined functions in time, ensuring continuity and differentiability for smoothness and curvature at any time domain of interest. To fully describe these functions explicitly and via a low-complexity algorithm, the proposed technique produces smooth, continuous, and differentiable trajectories of the spacecraft in CR3BP for any time domain of interest. Once these trajectories are fully determined, the position, velocity, and acceleration data of any object at any time are retrieved, even in continuous time changes. In this project, two types of complexities are considered: arithmetic and time complexities. The arithmetic complexity is determined by the number of floating-point operations required to execute the algorithm, whereas the time complexity is primarily related to the GPU execution time. These two types of complexities are interdependent, and any reduction in arithmetic complexity leads to a decrease in the GPU execution time. As demonstrated next, the proposed algorithm significantly reduces both complexities.

Let us consider a scenario where the position, velocity, and acceleration vectors, i.e., $\bar{x}(t_i), \dot{\bar{x}}(t_i), \ddot{\bar{x}}(t_i)$ in \mathbb{R}^3 , of the spacecraft are known at time t_i for $i = 0, 1, \dots, n$ and $t_0 < t_1 < \dots < t_n$. Thus, the trajectories of the spacecraft over n time intervals are described via n piecewise-defined functions. Moreover, these functions are defined at each interval $[t_k, t_{k+1}]$, where $k = 0, 1, \dots, n - 1$, and from \mathbb{R}^3 to \mathbb{R} such that the position functions (i.e. trajectories of the spacecraft) are expressed via

$$G_k(x(t)) = g_{0,k} + g_{1,k}t + g_{2,k}t^2 + g_{3,k}t^3 + g_{4,k}t^4 + g_{5,k}t^5, \tag{2}$$

and the corresponding velocity and acceleration functions of the spacecraft at each given time interval, respectively, are denoted via,

$$\begin{aligned} \dot{G}_k(x(t)) &= g_{1,k} + 2g_{2,k}t + 3g_{3,k}t^2 + 4g_{4,k}t^3 + 5g_{5,k}t^4, \\ \ddot{G}_k(x(t)) &= 2g_{2,k} + 6g_{3,k}t + 12g_{4,k}t^2 + 20g_{5,k}t^3 \end{aligned} \tag{3}$$

where $t_k \leq t \leq t_{k+1}$, $x(t)$ is the dimensionless quantity of the vector $\bar{x}(t)$, and $g_{0,k}, g_{1,k}, \dots, g_{5,k}$ are dimensionless quantities depend on the position, velocity, and acceleration of the spacecraft at each n time intervals. To accurately describe the trajectories of the spacecraft over n time intervals, it is necessary to explicitly determine the variables $g_{0,k}, g_{1,k}, \dots, g_{5,k}$ for n time intervals. To do so, one has to solve a system of $6n$ equations. The brute-force method of calculating these variables in each time interval followed by n intervals yields the explicit equations describing the spacecraft's trajectories at any time t with $\mathcal{O}(n^3)$ complexity. Therefore, a low-complexity algorithm is proposed that requires only $\mathcal{O}(n^2)$ operations, and hence to obtain these variables and describe the spacecraft's trajectories at any time.

To achieve this, the following technique for determining the spacecraft's trajectories is suggested. To begin with, the known vectors at the specific time values t_k and t_{k+1} are utilized, which serve as the boundary conditions for the $(k + 1)$ th time interval. These are expressed as:

$$\begin{aligned}
 G_k(x(t_k)) &= x(t_k) \\
 G_k(x(t_{k+1})) &= x(t_{k+1}) \\
 \dot{G}_k(x(t_k)) &= \dot{x}(t_k) \\
 \dot{G}_k(x(t_{k+1})) &= \dot{x}(t_{k+1}) \\
 \ddot{G}_k(x(t_k)) &= \ddot{x}(t_k) \\
 \ddot{G}_k(x(t_{k+1})) &= \ddot{x}(t_{k+1}),
 \end{aligned}
 \tag{4}$$

where $x(t_k)$, $\dot{x}(t_k)$, and $\ddot{x}(t_k)$ are position, velocity, and acceleration dimensionless quantities of the vectors $\bar{x}(t_k)$, $\dot{\bar{x}}(t_k)$, and $\ddot{\bar{x}}(t_k)$, respectively. Hence, within the time interval $[t_k, t_{k+1}]$, the motion of the spacecraft is expressed, as illustrated in Equations (2) and (3) adhering the boundary conditions Eq. (4). This allows to rewrite the set of equations as a matrix equation in the interval $[t_k, t_{k+1}]$ (note that there are n time intervals) s.t.

$$\underbrace{\begin{bmatrix} 1 & t_k & t_k^2 & t_k^3 & t_k^4 & t_k^5 \\ 1 & t_{k+1} & t_{k+1}^2 & t_{k+1}^3 & t_{k+1}^4 & t_{k+1}^5 \\ 0 & 1 & 2t_k & 3t_k^2 & 4t_k^3 & 5t_k^4 \\ 0 & 1 & 2t_{k+1} & 3t_{k+1}^2 & 4t_{k+1}^3 & 5t_{k+1}^4 \\ 0 & 0 & 2 & 6t_k & 12t_k^2 & 20t_k^3 \\ 0 & 0 & 2 & 6t_{k+1} & 12t_{k+1}^2 & 20t_{k+1}^3 \end{bmatrix}}_{A_k} \underbrace{\begin{bmatrix} g_{0,k} \\ g_{1,k} \\ g_{2,k} \\ g_{3,k} \\ g_{4,k} \\ g_{5,k} \end{bmatrix}}_{\underline{x}_k} = \underbrace{\begin{bmatrix} x(t_k) \\ x(t_{k+1}) \\ \dot{x}(t_k) \\ \dot{x}(t_{k+1}) \\ \ddot{x}(t_k) \\ \ddot{x}(t_{k+1}) \end{bmatrix}}_{\underline{b}_k}.
 \tag{5}$$

We note here that a brute-force calculation in determining the vectors \underline{b}_k for all k 's over n time intervals cost $\mathcal{O}(n^3)$ operations. More specifically, the unknown trajectory coefficients for the $(k + 1)$ th time interval, i.e., $[t_k, t_{k+1}]$ are obtained by taking the explicit inverse of A_k and multiplying it with the boundary conditions vector \underline{b}_k . Hence, to fully solve the system this process needs to be repeated for each subsequent time interval, until all n time intervals have been considered, thereby fully determining the satellite's trajectories at any given time. However, as mentioned earlier this approach is computationally expensive, and more specifically the explicit inverse of dense matrices is rarely computed [34]. To mitigate this and reduce the complexity, LU decomposition is used to factor the coefficient matrix A_k in the system of Eq. (5). Then, the bidiagonal lower triangular matrices \tilde{L}_r are used s.t. $\tilde{L}_r = L_r^{-1}$ to compute the matrix–vector product from $\tilde{L}_1 \underline{b}_k$ and multiplying the resultant vector in order $\tilde{L}_2, \dots, \tilde{L}_5$ followed by the backward substitution to reduce the computational burden. By adopting this approach, the trajectory coefficients are obtained for all n time intervals having $\mathcal{O}(n^2)$ as opposed to $\mathcal{O}(n^3)$ complexity algorithm. Let us, therefore, utilize the LU decomposition of the coefficient matrix A_k in Eq. (5):

$$\left(\prod_{r=1}^5 L_r \right) U_k \underline{x}_k = \underline{b}_k, \text{ where } A_k = \left(\prod_{r=1}^5 L_r \right) U_k
 \tag{6}$$

where $L_r \in \mathbb{R}^{6 \times 6}$, $r = 1, 2, \dots, 5$ are bidiagonal lower triangular matrices, and $U_k \in \mathbb{R}^{6 \times 6}$ is an upper triangular matrix. Moreover, these bidiagonal and upper triangular matrices are explicitly given via

$$\begin{aligned}
 L_1 &= \begin{bmatrix} 1 & & & & & \\ 1 & d_k & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 & d_k \end{bmatrix}, & L_2 &= \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & -d_k & & \\ & & & & 1 & \\ & & & & & 1 & -d_k \\ & & & & & & 1 \end{bmatrix}, \\
 L_3 &= \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 2 & d_k & \\ & & & & 1 & \\ & & & & & 2 & d_k \end{bmatrix}, \\
 L_4 &= \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 3 & -d_k \\ & & & & & 1 \end{bmatrix}, \\
 L_5 &= \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 2 & d_k \end{bmatrix}, & U_k &= \begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ & 1 & c_{1,k} & c_{2,k} & c_{3,k} & c_{4,k} \\ & & 1 & e_{1,k} & e_{2,k} & e_{3,k} \\ & & & 1 & 2c_{1,k} & f_k \\ & & & & 2 & 2m_k \\ & & & & & 2 \end{bmatrix}
 \end{aligned} \tag{7}$$

where the pre-computed entries are given by

$$\begin{aligned}
 d_k &= t_{k+1} - t_k, \\
 c_{1,k} &= t_{k+1} + t_k, \\
 c_{2,k} &= t_{k+1}^2 + t_{k+1}t_k + t_k^2, \\
 c_{3,k} &= t_{k+1}^3 + t_{k+1}^2t_k + t_{k+1}t_k^2 + t_k^3, \\
 c_{4,k} &= t_{k+1}^4 + t_{k+1}^3t_k + t_{k+1}^2t_k^2 + t_{k+1}t_k^3 + t_k^4, \\
 e_{1,k} &= t_{k+1} + 2t_k, \\
 e_{2,k} &= t_{k+1}^2 + 2t_{k+1}t_k + 3t_k^2, \\
 e_{3,k} &= t_{k+1}^3 + 2t_{k+1}^2t_k + 3t_{k+1}t_k^2 + 4t_k^3, \\
 f_k &= 3t_{k+1}^2 + 4t_{k+1}t_k + 3t_k^2, \\
 m_k &= 2t_{k+1} + 3t_k.
 \end{aligned} \tag{8}$$

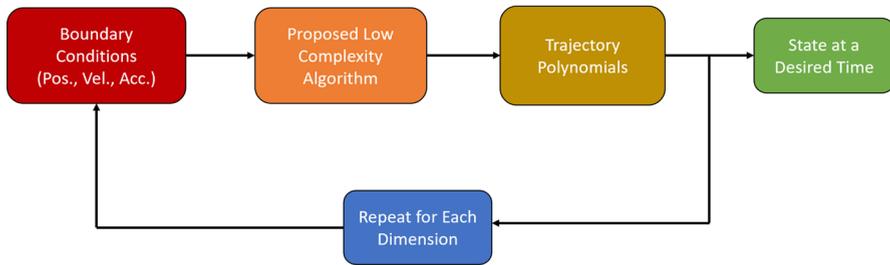


Fig. 4 Propagation of each orbit using respective methods

The subscript k corresponds to the $(k + 1)$ th time interval which is $[t_k, t_{k+1}]$, and the empty spaces of the matrices represent zero elements. Even for all n time intervals, the pre-computation cost of the entries in Eq. (8) cost $\mathcal{O}(n)$ operations as it is an update of the consecutive vectors. After obtaining $g_{0,k}, \dots, g_{5,k}$ for $k = 0, 1, \dots, n - 1$, the trajectory of the spacecraft can fully be described via Eq. (3) over n time intervals using $\mathcal{O}(n^2)$ as opposed to a $\mathcal{O}(n^3)$ algorithm. Furthermore, to ensure the reliability of this algorithm, refer to matrix decomposition techniques that have been successfully applied in trajectory determination for a quadcopter [16] and a servicing robotic arm [17]. In conclusion, these trajectories serve as reference trajectories at any given time, and can even be extrapolated to predict the future motion of the spacecraft.

4 Results

This section shows the accuracy and time complexity of the proposed algorithm while comparing it with the ODE45 which is a well-known, traditional, and accurate orbit propagator. The analysis is performed for a LLO, NRHO, DRO and a Lyapunov orbit. The proposed low-complexity algorithm first requires the position, velocity, and acceleration of the six boundary conditions. Then, the conditions are inputted into the algorithm and the algorithm produces the polynomial that represents the motion of the object. Finally, the process is repeated for the remaining dimensions, and the state at a desired time is found using the generated polynomials. An overview of the process is shown in Fig. 4. The algorithm is tested on a number of important orbit trajectories within the Cislunar region using the CR3BP model. The Artemis I mission traveled in a partial DRO around the Moon at the end of 2022, and the complete DRO will be the first trajectory of interest [7]. Next, low-Lunar orbits are orbits with very low altitudes over the Moon, resulting in motion very similar to two-body motion as the Moon's gravity dominates the spacecraft's trajectory. Missions such as KPLO and other Lunar surface surveying missions will fly in low-Lunar orbits, marking the second trajectory of interest [4]. The third trajectory to be tested will be an L_2 Lyapunov orbit. Finally, NASA's Gateway station will be a vital location for missions in the Cislunar region, acting as a staging point for Lunar surface missions [35]. Gateway will orbit the Moon in a southern NRHO, identifying

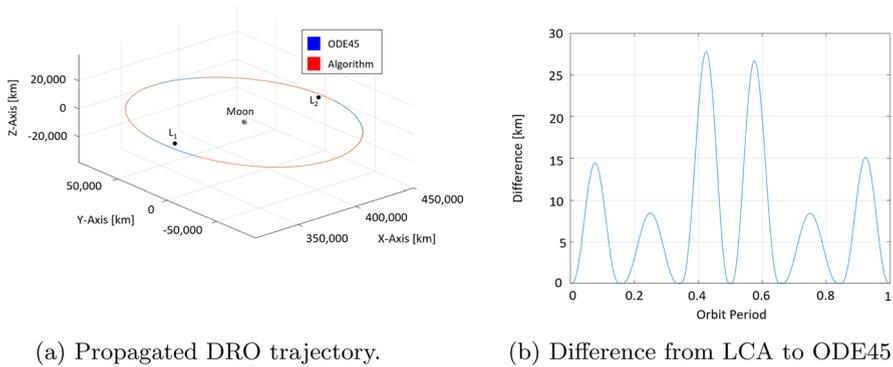


Fig. 5 Distant retrograde orbit (DRO) analysis

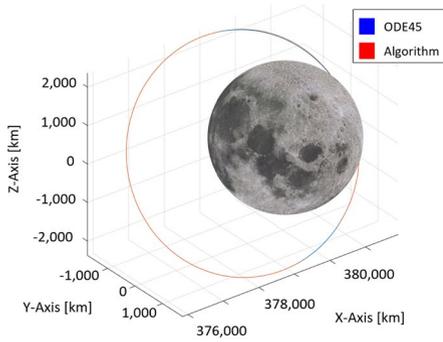
the fourth trajectory of interest. The algorithm is completed using seven boundary conditions, marking six time intervals of a desired trajectory, over one orbit period. The initial conditions of the trajectory are known, and the periodic orbits are originally propagated using ODE45. The boundary conditions are then retrieved from the ODE45 results and used in the low-complexity algorithm (LCA). Finally, the resulting trajectories are represented in the dimensional CR3BP rotating frame.

The DRO orbit is propagated using the LCA and compared to the ODE45 results. Figure 5a shows the DRO orbit constructed using both ODE45 and the LCA. The trajectories are difficult to distinguish because of how close the algorithm’s and ODE45’s trajectories are to one another. Therefore, the difference in position between the LCA and the ODE45 trajectory for the DRO is shown in Fig. 5b. Since the algorithm must meet the boundary conditions outlined by ODE45, the intervals in which the algorithm is broken into are typically distinguished by a difference of 0 between the LCA and ODE45 trajectory. The algorithm matches the ODE45 resolved trajectory extremely closely, deviating up to 27 km. A difference of 27 kms between ODE45 and the LCA is insignificant in reference to the 100,000 kms the DRO stretches across.

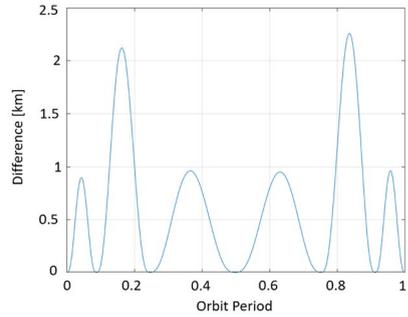
The analysis between the LCA and ODE45 is completed for LLO next. For the analysis, a sample LLO trajectory is used with a perilune of 100 kms. The resultant trajectories are seen in Fig. 6a and the difference between the LCA and ODE45 trajectories is presented in Fig. 6b. The LLO has a peak difference of 2.3 kms occurring in the fifth time interval of the algorithm. The 2.3 kms difference is also insignificant even though the orbit traverses a smaller distance of around 1000 kms.

The L_2 Lyapunov orbit constructed by the LCA is presented in Fig. 7a, and the difference from the ODE45 propagation is observed in Fig. 7b. The LCA’s model of the sample Lyapunov orbit has the closest resemblance to the ODE45 trajectory out of all the tested orbits. The peak difference occurs in the third time interval set by the boundary conditions, only reaching right over 2.5 kms.

Finally, Fig. 8a shows the LCA’s model of the NRHO orbit and Fig. 8b conveys the difference of the LCA and ODE45 propagated trajectories for the NRHO orbit. Out of the selected sample trajectories, the NRHO is found to be the most

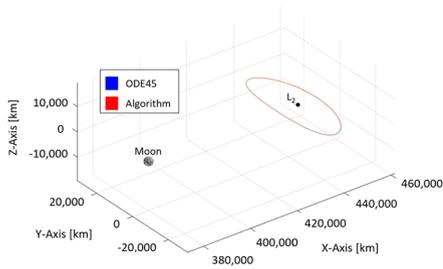


(a) Propagated LLO trajectory.

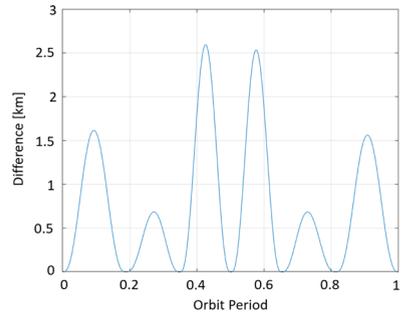


(b) Difference from LCA to ODE45.

Fig. 6 Low-Lunar orbit (LLO) analysis

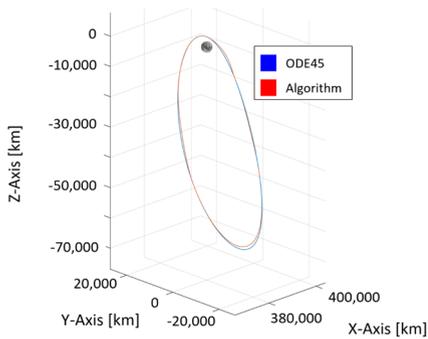


(a) Propagated Lyapunov trajectory.

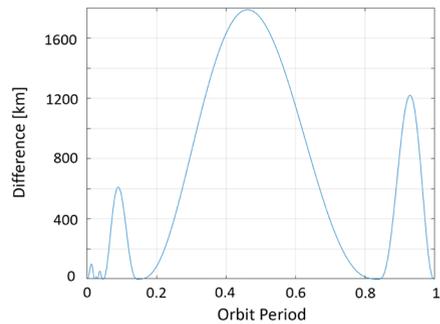


(b) Difference from LCA to ODE45.

Fig. 7 L_2 Lyapunov analysis



(a) Propagated NRHO trajectory.



(b) Difference from LCA to ODE45.

Fig. 8 Near-rectilinear halo orbit (NRHO) analysis

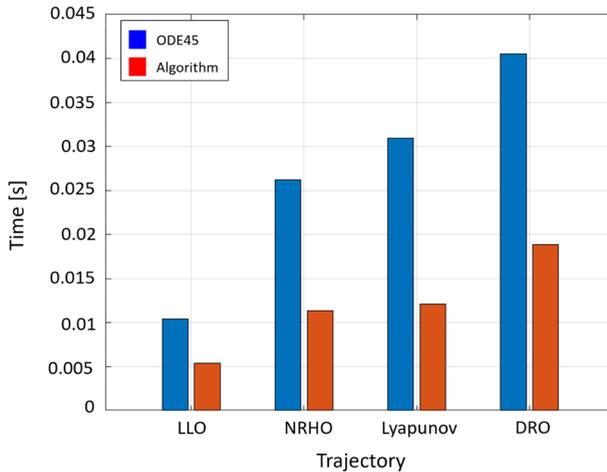


Fig. 9 Propagation time of each orbit using respective methods

challenging to construct accurately using the LCA. The NRHO is the only trajectory in which the ODE45 and LCA models can be distinguished from one another at certain times, as seen in Fig. 8a. The NRHO has the largest difference from the ODE45 path, peaking at about 1800 kms. The large difference is partially a result of how the boundary conditions are spread throughout the trajectory. Points in the trajectory that change rapidly, such as perilune, have tighter spaced boundary conditions to prevent drastic changes in the algorithm. Alternatively, the boundary conditions are more spread out through points in the NRHO where conditions change more slowly, such as apolune, so the algorithm has more of a gradual change over a large period of time. This is shown in Fig. 8a and b, in which the trajectory begins at perilune with low deviations and slowly begins to deviate, reaching the peak difference at apolune, halfway through the orbit.

The drive for producing an LCA is to reduce computational time and arithmetic complexity and to address the physical requirements for smooth trajectory determination. Typically, arithmetic complexity is proportional to the time complexity of a given algorithm. Therefore, the computational time (i.e. time complexity) of the algorithm is determined for each of the four trajectories and compared to the computational time of ODE45. The time both methods take to resolve a trajectory can slightly vary based on rounding and computer performance during the propagation. As a result, the computed times shown in Fig. 9 are the average computational time of 100 propagations of each respective trajectory. The reductions in computational time of the algorithm are highlighted in Fig. 9 as it depicts that the algorithm resolves the trajectory significantly faster than ODE45, clocking in at about half of the time for each orbit.

The selection of the initial and boundary conditions is paramount when using the low-complexity algorithm. For the sample trajectories above, the conditions are selected to highlight the algorithm's ability to accurately reconstruct common orbits. This proved challenging for some orbits as changing the initial conditions

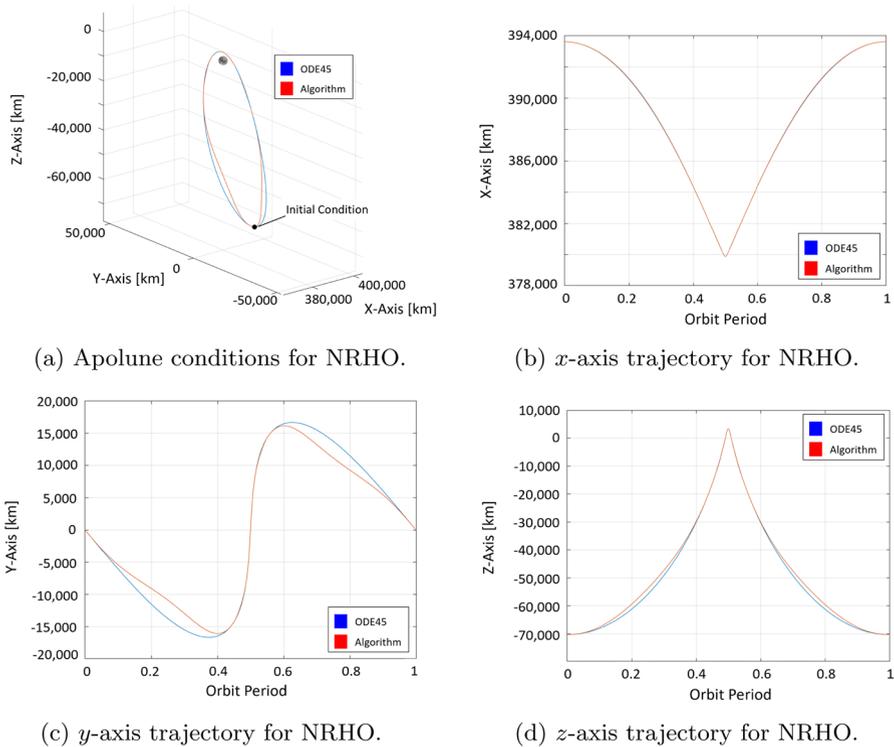


Fig. 10 Apolune NRHO trajectory analysis

of sensitive orbits, such as the NRHO, would drastically affect the LCA’s ability to reconstruct the orbit. To demonstrate this effect, take the NRHO example. The NRHO constructed in Fig. 8a used initial conditions near the perilune of the orbit, with frequent boundary conditions near perilune. The reconstruction is slightly inaccurate from ODE45, but generally, the orbit retained its shape. Moving the initial conditions of the problem to around the apolune of the NRHO, with boundary conditions still frequenting perilune, the changes in the trajectory are seen in Fig. 10a. The trajectory generated by the LCA clearly propagates the NRHO incorrectly near apolune, in a more pronounced manner than in Fig. 8a. For a closer examination of the problem, the LCA and ODE45 trajectory is plotted over the x -axis, y -axis, and z -axis (Fig. 10b–d). The algorithm is completed in one-dimension at a time and as a result, the accuracy of the NRHO trajectory varies depending on the axis. In this example, the x -axis and z -axis trajectories have accurate results, but y -axis trajectory deviates more significantly. The clear effect changing the initial conditions of the orbit has on the resultant trajectory highlights the importance of the selecting informative initial conditions for the low-complexity algorithm.

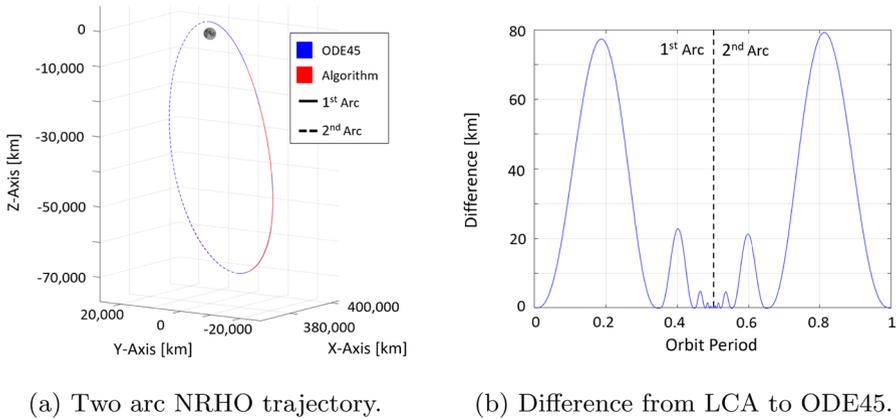


Fig. 11 Two arc NRHO analysis

Two factors contribute to producing inaccuracy of the LCA, poor selection of the initial conditions and the spacing of the boundary conditions. These factors are presented in the application of the LCA in determining the NRHO (Figs. 8a, 10a). The spacing of boundary conditions resulted in inaccuracy because the algorithm concentrated boundary conditions near intervals of rapid change, allowing for intervals of slow change to slowly deviate. The spacing of the boundary conditions can be resolved by cutting down the length of the trajectory the algorithm must solve for at a time. Previously, the algorithm completed one orbit of the NRHO, now the NRHO will be split into two halves and ran twice to complete the trajectory. Using the apolune initial conditions, Fig. 11a shows the LCA and ODE45 trajectories for the NRHO propagated in two arcs and Fig. 11b shows the difference between the two propagation methods. Breaking the trajectory into two arcs resolves the issue of the LCA being unable to produce the NRHO trajectory. The maximum difference between the LCA and ODE45 propagation drops from 1800 kms (Fig. 8b) to 78 kms (Fig. 11b). Additionally, the initial conditions at the apolune of the NRHO previously generated the less accurate NRHO, but is now able to produce an accurate trajectory. The computational time of the NRHO in two arcs is compared to the propagation time of the NRHO using ODE45 and the LCA in a single arc, shown in Fig. 12. The results show that completing two arcs as opposed to a single arc increases the LCA computation by around 0.005 s. Overall, by breaking the trajectory up, the LCA is able to produce an accurate trajectory for the NRHO orbit with some additional computation time. The method of breaking the trajectory into smaller arcs can be extended further to troubleshooting other trajectories that are inaccurate over a given time interval, as completed for the NRHO.

The selection of boundary conditions used in the algorithm directly impact the results and accuracy of the LCA. Generally, for an orbit to be discretized into different arcs that accurately represent the numerically propagated trajectory, the boundary conditions must be selected to create short intervals during sensitive portions of the trajectory. For the selected orbits, the sensitive portions of the trajectory are

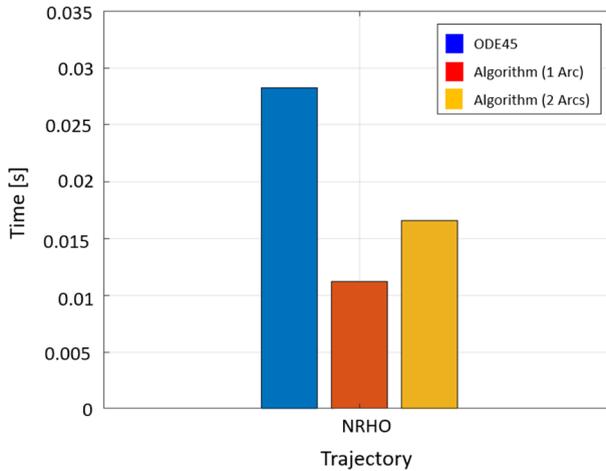


Fig. 12 Computation time of NRHO with apolune initial conditions

points of rapid change, occurring at the locations in space where the orbit is the closest to the Moon. This is seen clearly in the NRHO and LLO examples. Both trajectories start at perilune, with the boundary conditions being needed slightly closer in time at the beginning and end of the orbital periods. In contrast, the DRO is generally the same distance from the Moon and, therefore, the spacing of boundary conditions are uniform. Furthermore, in the instance that the selected discretized arcs do not meet desired accuracy, additional boundary conditions are implemented to further limit the time between boundaries. Once again this is demonstrated in the NRHO case, in which the original process of discretizing the orbit into arcs was unsatisfactory, and thus additional boundary conditions were required to create an accurate representation without significantly impacting computation time.

For efficient use of the LCA in the CR3BP, a relationship between the dynamics of a trajectory and the performance of LCA must be established. The stability of two orbits as well as their associated performance in the LCA are examined. The first selected trajectory is the L_2 Lyapunov orbit, a trajectory showing promising performance of the LCA, and the second trajectory is a new L_2 Lyapunov orbit. The stability of both orbits is determined using the eigenvalues of the monodromy matrix for each periodic orbit [36]: a saddle mode exists when the eigenvalue has one positive and one negative real part; otherwise, a center mode exists when the eigenvalue has complex conjugates. The original Lyapunov orbit possesses one saddle and two center modes, while the new Lyapunov orbit possesses two saddles and one center mode. The new Lyapunov orbit (Fig. 13a) is initiated for propagation at apolune, similar to the original Lyapunov. The comparison between LCA and ODE45 is represented in Fig. 13b. The first Lyapunov orbit outgrows the performance of the second one. These results possibly indicate that the stability of an orbit is associated to its performance in the LCA. Additionally, the previously analyzed NRHO is found to have one saddle and two center modes, similar to the original Lyapunov orbit. The previous finding has demonstrated that the NRHO yields unsatisfactory

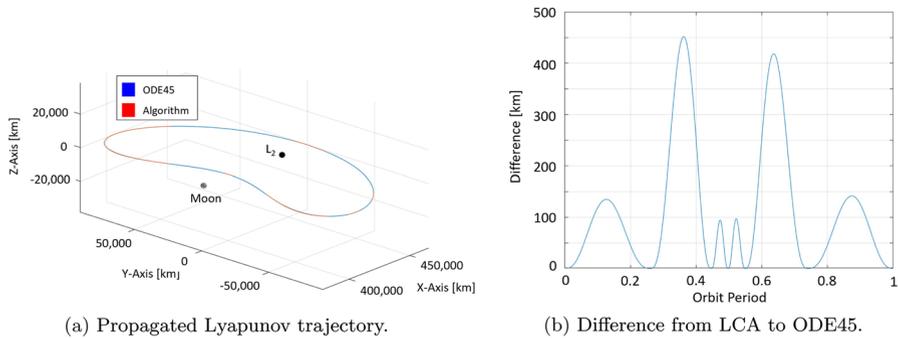


Fig. 13 Second L_2 Lyapunov for further analysis of LCA performance

outcomes compared to the original Lyapunov orbit, despite possessing similar stability properties, but significantly differing in LCA performance. Consequently, the suggested comparison of two Lyapunov orbits with distinct eigenvalues cannot fully establish that a stability relationship exists between dynamics and the performance of LCA. Therefore, future research will delve into exploring the connection between trajectory properties and LCA performance in greater depth.

5 Conclusions

Analytical solutions to predict trajectories in complex, multi-body dynamic systems, such as the CR3BP, may be computationally expensive to solve iteratively. The expense of iterative methods such as Gauss–Legendre, Dormand–Prince, and Chebyshev–Picard, warrants the need for lower complexity algorithms, specially for on-board navigation purposes. The process stems from the Lagrangian interpolation, fitting curves to given conditions to analytically solve for trajectories. The trajectories are extended from linear fitting to a more accurate polynomial in time solution. Furthermore, interpolation methods can still be computationally expensive when using brute force methods to solve for the polynomial, requiring $\mathcal{O}(n^3)$ operations. The designed low-complexity algorithm takes advantage of the low computational necessities of interpolation and the accuracy of a polynomial fit to create an algorithm that requires $\mathcal{O}(n^2)$ operations.

The low-complexity algorithm is tested by propagating well-known trajectories in the CR3BP and comparing the motion to the results of an ODE45 propagated trajectory. The low-complexity algorithm accurately represented a DRO, LLO, and L_2 Lyapunov orbit, with a significantly lower computational time than ODE45. The NRHO initially challenged the algorithm to produce an accurate trajectory due to the diverse motion of the orbit. The selected boundary conditions allowed for accurate motion during intervals of rapid change in the NRHO and slow accumulating deviation in intervals of slow change. Breaking the NRHO into two arcs and running the algorithm twice lowers the time between intervals and allowed for the algorithm to produce accurate motion of the NRHO. Overall, the proposed low-complexity

algorithm offers a lower computational (arithmetic and time) method for trajectory propagation. In comparison to ODE45, the algorithm computes accurate trajectories for an array of CR3BP orbits in a significantly lower time. The efficiency and reliability of the proposed algorithm make it for a potential resource for on-board trajectory propagation in the Cislunar region. Throughout this work, the trajectories generated by the LCA are only utilized to interpolate between the provided boundary conditions. Future development entails the testing of the algorithm's ability to predict trajectories outside of the boundary conditions.

Appendix A: Differential Corrections to Obtain Periodic Orbits in the CR3BP

Spacecraft trajectory design typically relies on numerical strategies such as differential corrections to fulfill mission requirements by targeting specific end conditions while varying a set of initial conditions. Consequently, it is important to find a correlation between the variations in the initial state of a trajectory, $\delta\bar{x}_0$, with the variations of its final states downstream, $\delta\bar{x}_f$:

$$\delta\bar{x}_f = \frac{\partial\bar{x}(t_f)}{\partial\bar{x}(t_0)}\delta\bar{x}_0, \quad (9)$$

where t_0 and t_f are the initial and final times along the propagated trajectory, respectively. Note that overbars denote vectors. The vector \bar{x} is a six-dimensional vector that contains position and velocity states of a s/c. Therefore, $\delta\bar{x}(t) = [\delta x, \delta y, \delta z, \delta \dot{x}, \delta \dot{y}, \delta \dot{z}]^T$ (the subscript "T" denotes the transpose). The term $\frac{\partial\bar{x}(t_f)}{\partial\bar{x}(t_0)}$ in Eq. (9) is a variable sensitivity matrix very useful for targeting schemes and stability analysis. Such a matrix is also denoted as the State Transition Matrix (STM), $\phi(t_f, t_0)$, and it is associated with the variational equations relative to any general reference arc that satisfies any non-linear differential equations:

$$\phi(t_f, t_0) = \frac{\partial\bar{x}(t_f)}{\partial\bar{x}(t_0)}. \quad (10)$$

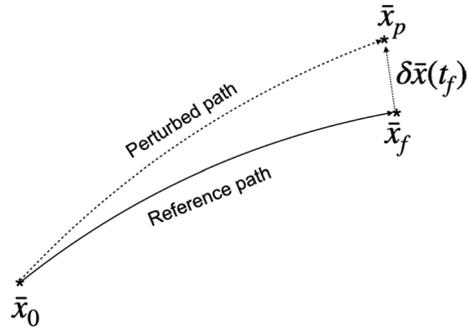
Figure 14 illustrates a reference trajectory arc between \bar{x}_0 and \bar{x}_f , along with a perturbed path \bar{x}_p relative to the reference, requiring modification of the initial state on the reference path to reach the desired position \bar{x}_p , considering the first-order form of the non-linear system of differential equations:

$$\dot{\bar{x}} = \bar{f}(\bar{x}, t), \quad (11)$$

where dots indicate the derivative with respect to time, then \bar{x}_p at t_f is represented as a Taylor series expansion relative to the reference. The linear variational equations, derived from the equations of motion, are provided in the form:

$$\delta\dot{\bar{x}}(t) = \mathbf{A}(t)\delta\bar{x}(t). \quad (12)$$

Fig. 14 Perturbed path relative to the reference path of a trajectory



$\mathbf{A}(t)$ is the Jacobian matrix comprised of the partials of the equations of motion with respect to the states evaluated at time t :

$$\mathbf{A}(t) = \frac{\partial \vec{f}(\vec{x}, t)}{\partial \vec{x}(t)}. \tag{13}$$

The solution to Eq. (12) is the previously presented Eq. (9). The evolution of $\phi(t, t_0)$ is governed by the following matrix differential equation [36]:

$$\dot{\phi}(t, t_0) = \mathbf{A}(t)\phi(t, t_0), \tag{14}$$

which represents 36 scalar differential equations, given that $\phi(t, t_0)$ is a 6x6 matrix. In the CR3BP, Eq. (14) holds as:

$$\begin{pmatrix} \delta \dot{x} \\ \delta \dot{y} \\ \delta \dot{z} \\ \delta \ddot{x} \\ \delta \ddot{y} \\ \delta \ddot{z} \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ U_{xx}^* & U_{xy}^* & U_{xz}^* & 0 & 2 & 0 \\ U_{xy}^* & U_{yy}^* & U_{yz}^* & -2 & 0 & 0 \\ U_{xz}^* & U_{yz}^* & U_{zz}^* & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} \delta x \\ \delta y \\ \delta z \\ \delta \dot{x} \\ \delta \dot{y} \\ \delta \dot{z} \end{pmatrix} \tag{15}$$

where the 3×3 block at the bottom-left of the matrix corresponds to the second partial derivatives of U^* [37]. The partial derivatives within the STM reflect the impact of changes in initial state elements on the corresponding final state elements:

$$\begin{pmatrix} \delta x_f \\ \delta y_f \\ \delta z_f \\ \delta \dot{x}_f \\ \delta \dot{y}_f \\ \delta \dot{z}_f \end{pmatrix} = \begin{bmatrix} \frac{\partial x}{\partial x_0} & \frac{\partial x}{\partial y_0} & \frac{\partial x}{\partial z_0} & \frac{\partial x}{\partial \dot{x}_0} & \frac{\partial x}{\partial \dot{y}_0} & \frac{\partial x}{\partial \dot{z}_0} \\ \frac{\partial y}{\partial x_0} & \frac{\partial y}{\partial y_0} & \frac{\partial y}{\partial z_0} & \frac{\partial y}{\partial \dot{x}_0} & \frac{\partial y}{\partial \dot{y}_0} & \frac{\partial y}{\partial \dot{z}_0} \\ \frac{\partial z}{\partial x_0} & \frac{\partial z}{\partial y_0} & \frac{\partial z}{\partial z_0} & \frac{\partial z}{\partial \dot{x}_0} & \frac{\partial z}{\partial \dot{y}_0} & \frac{\partial z}{\partial \dot{z}_0} \\ \frac{\partial \dot{x}}{\partial x_0} & \frac{\partial \dot{x}}{\partial y_0} & \frac{\partial \dot{x}}{\partial z_0} & \frac{\partial \dot{x}}{\partial \dot{x}_0} & \frac{\partial \dot{x}}{\partial \dot{y}_0} & \frac{\partial \dot{x}}{\partial \dot{z}_0} \\ \frac{\partial \dot{y}}{\partial x_0} & \frac{\partial \dot{y}}{\partial y_0} & \frac{\partial \dot{y}}{\partial z_0} & \frac{\partial \dot{y}}{\partial \dot{x}_0} & \frac{\partial \dot{y}}{\partial \dot{y}_0} & \frac{\partial \dot{y}}{\partial \dot{z}_0} \\ \frac{\partial \dot{z}}{\partial x_0} & \frac{\partial \dot{z}}{\partial y_0} & \frac{\partial \dot{z}}{\partial z_0} & \frac{\partial \dot{z}}{\partial \dot{x}_0} & \frac{\partial \dot{z}}{\partial \dot{y}_0} & \frac{\partial \dot{z}}{\partial \dot{z}_0} \end{bmatrix} \begin{pmatrix} \delta x_0 \\ \delta y_0 \\ \delta z_0 \\ \delta \dot{x}_0 \\ \delta \dot{y}_0 \\ \delta \dot{z}_0 \end{pmatrix} \tag{16}$$

Using the STM is crucial in targeting schemes as it serves as an effective linear predictor of the sensitivity of the final state to variations in the initial state. Iteratively updating initial conditions based on the STM and terminal constraint sensitivities, it is possible to enable the determination of periodic orbits using a multi-variable Newton–Raphson strategy. Consider that $\bar{\mathcal{Y}}$ contains the design variables of the problem (or free variable) and $\bar{\mathcal{F}}(\bar{\mathcal{Y}})$ the terminal constraints:

$$\bar{\mathcal{Y}} = (\mathcal{Y}_1, \mathcal{Y}_2, \dots, \mathcal{Y}_n), \quad \bar{\mathcal{F}}(\bar{\mathcal{Y}}) = (\mathcal{F}_1(\bar{\mathcal{Y}}), \mathcal{F}_2(\bar{\mathcal{Y}}), \dots, \mathcal{F}_j(\bar{\mathcal{Y}})). \tag{17}$$

The objective is to determine the vector $\bar{\mathcal{Y}}$ that nullifies the constraint vector $\bar{\mathcal{F}}(\bar{\mathcal{Y}}) = \bar{0}$, with convergence measured by the norm of $\bar{\mathcal{F}}$ being smaller than a user-defined accuracy ϵ , set to $\epsilon = 10^{-12}$, and the update equation for the Newton–Raphson algorithm involving a first-order Taylor series expansion about the initial condition $\bar{\mathcal{Y}}_0$ [38]:

$$\bar{\mathcal{F}}(\bar{\mathcal{Y}}) \approx \bar{\mathcal{F}}(\bar{\mathcal{Y}}_0) + D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)(\bar{\mathcal{Y}} - \bar{\mathcal{Y}}_0) = 0, \tag{18}$$

where $D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)$ contains the partial derivatives of the constraints with respect to the free variables:

$$D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0) = \frac{\partial \bar{\mathcal{F}}(\bar{\mathcal{Y}})_0}{\partial \bar{\mathcal{Y}}_0} = \begin{bmatrix} \frac{\partial \mathcal{F}_1}{\partial \mathcal{Y}_1} & \frac{\partial \mathcal{F}_1}{\partial \mathcal{Y}_2} & \dots & \frac{\partial \mathcal{F}_1}{\partial \mathcal{Y}_n} \\ \frac{\partial \mathcal{F}_2}{\partial \mathcal{Y}_1} & \frac{\partial \mathcal{F}_2}{\partial \mathcal{Y}_2} & \dots & \frac{\partial \mathcal{F}_2}{\partial \mathcal{Y}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathcal{F}_j}{\partial \mathcal{Y}_1} & \frac{\partial \mathcal{F}_j}{\partial \mathcal{Y}_2} & \dots & \frac{\partial \mathcal{F}_j}{\partial \mathcal{Y}_n} \end{bmatrix}_0 \tag{19}$$

The method to solve the equation depends on the number of constraints relative to the number of free variables. In the case where the number of constraints is equal to the number of free variables and the matrix $D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)$ is invertible, a unique solution is obtained through the iterative process:

$$\bar{\mathcal{Y}} = \bar{\mathcal{Y}}_0 - D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)^{-1} \bar{\mathcal{F}}(\bar{\mathcal{Y}}). \tag{20}$$

On the other hand, when the number of constraints is less than the number of free variables, and $D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)$ is not invertible, there are infinitely many solutions, with the solution closest to the final time control variable vector given by the minimum-norm solution:

$$\bar{\mathcal{Y}} = \bar{\mathcal{Y}}_0 - D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)^T [D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)D\bar{\mathcal{F}}(\bar{\mathcal{Y}}_0)^T]^{-1} \bar{\mathcal{F}}(\bar{\mathcal{Y}}). \tag{21}$$

Each periodic orbit has specific conditions and constraints that need to be satisfied, and the computed $\bar{\mathcal{Y}}$ is assessed to see if the norm of $\bar{\mathcal{F}}$ is within the defined tolerance ϵ , with the process repeated until convergence is achieved.

Data Availability The datasets analysed during the current study are referenced in the text and publicly available.

References

1. Balossino, A., Davarian, F.: The plan to give the moon decent wireless coverage. *IEEE Spectrum* **59**, 32 (2022)
2. Baker-McEvelly, B., Doroba, S., Gilliam, A., Criscola, F., Canales, D., Frueh, C., Henderson, T.: A review on hot-spot areas within the cislunar region and upon the moon surface, and methods to gather passive information from these regions. In: *AAS/AIAA 33rd Space Flight Mechanics Meeting* (2023)
3. Kazmerchuk, P.V., Shirshakov, A.E.: The Luna-25 spacecraft: return to the Moon. *Sol. Syst. Res.* **55**, 496–508 (2021). <https://doi.org/10.1134/S0038094621060058>
4. Song, Y.-J., Kim, Y.-R., Bae, J., Park, J.-I., Hong, S., Lee, D., Kim, D.-K.: Overview of the flight dynamics subsystem for Korea Pathfinder Lunar Orbiter mission. *Aerospace* **8**, 222 (2021). <https://doi.org/10.3390/aerospace8080222>
5. Ohtake, K.: Current status of planned lunar polar exploration mission jointly studied by India and Japan. In: *52nd Lunar and planetary science conference* (2021)
6. Johnson, K.: Fly me to the moon, worldwide cislunar and lunar missions. *CSIS Aerospace Security Project* (2022)
7. NASA's management of the Artemis missions. Office of The Inspector General, Office of Audits (2021). <https://www.oversight.gov/report/NASA/NASA%E2%80%99s-Management-Artemis-Missions>
8. Keller, H.B.: Numerical Solution of Two Point Boundary Value Problems. SIAM, Philadelphia (1976). <https://doi.org/10.1137/1.9781611970449>
9. Roberts, S.M., Shipman, J.S.: Continuation in shooting methods for two-point boundary value problems. *J. Math. Anal. Appl.* **18**(1), 45–58 (1967)
10. Pavlak, T., Howell, K.: Strategy for optimal, long-term stationkeeping of libration point orbits in the earth-moon system. In: *AIAA/AAS Astrodynamics Specialist Conference*, p. 4665 (2012)
11. Miao, Q., McDaid, A., Zhang, M., Kebria, P., Li, H.: A three-stage trajectory generation method for robot-assisted bilateral upper limb training with subject-specific adaptation. *Robot. Auton. Syst.* **105**, 38–46 (2018). <https://doi.org/10.1016/j.robot.2018.03.010>
12. Dhullipalla, M.H., Hamrah, R., Warier, R., Amit, S.: Trajectory generation on SE(3) for an underactuated vehicle with pointing direction constraints. In: *2019 American Control Conference (AAC)*, pp. 1930–1935 (2019). <https://doi.org/10.23919/ACC.2019.8815238>
13. Khalil, W., Dombre, E.: Modeling, Identification and Control of Robots. Kogan Page Science paper edition Modeling, identification & control of robots. Elsevier Science, Amsterdam (2004). <https://doi.org/10.1016/B978-1-903996-66-9.X5000-3>
14. Sandberg, A., Sands, T.: Autonomous trajectory generation algorithms for spacecraft slew maneuvers. *Aerospace* **9**, 135 (2022). <https://doi.org/10.3390/aerospace9030135>
15. Liu, H., Qu, D., Xu, F., Du, Z., Jia, K., Liu, M.: An efficient online trajectory generation method based on kinodynamic path search and trajectory optimization for human-robot interaction safety. *Entropy* (2022). <https://doi.org/10.3390/e24050653>
16. Kurttisi, A., Perera, S.M., Merve, D.: An analytical solution to determine polynomial trajectories and adaptive nonlinear dynamical inversion control of a quadcopter. In: *AIAA SCITECH 2023 Forum, AIAA 2023-0509* (2023). <https://doi.org/10.2514/6.2023-0509>
17. Malik, A., Perera, S.M., Henderson, T.: An efficient quintic time scaling end-effector trajectory generation algorithm for a servicing robotic arm. In: *AIAA SCITECH 2023 Forum, AIAA 2023-0699* (2023). <https://doi.org/10.2514/6.2023-0699>
18. Aristoff, J.M., Horwood, J.T., Poore, A.B.: Orbit and uncertainty propagation: a comparison of Gauss-Legendre-, Dormand-Prince-, and Chebyshev-Picard-based approaches. *Celest. Mech. Dyn. Astron.* **118**(1), 13–28 (2014). <https://doi.org/10.1007/s10569-013-9522-7>
19. Deuffhard, P.: Order and stepsize control in extrapolation methods. *Numer. Math.* **41**(3), 399–422 (1983). <https://doi.org/10.1007/BF01418332>
20. Süli, E., Mayers, D.F.: An Introduction to Numerical Analysis. Cambridge University Press, Cambridge (2003). <https://doi.org/10.1017/CBO9780511801181>
21. Wie, B.: Space vehicle dynamics and control, 2nd edition. AIAA (2008). <https://doi.org/10.2514/4.860119>
22. Melton, R.G.: Fundamentals of astrodynamics and applications. *J. Guid. Control Dyn.* **21**(4), 672–672 (1998). <https://doi.org/10.2514/2.4291>

23. Montenbruck, O., Gill, E., Lutze, F.: Satellite orbits: models, methods, and applications. *Appl. Mech. Rev.* **55**(2), 27–28 (2002). <https://doi.org/10.1115/1.1451162>
24. Montenbruck, O.: Numerical integration methods for orbital motion. *Celest. Mech. Dyn. Astron.* **53**, 59–69 (1992). <https://doi.org/10.1007/BF00049361>
25. Jones, B.A., Anderson, R.L.: A survey of symplectic and collocation integration methods for orbit propagation. In: 22nd AAS/AIAA Space Flight Mechanics Meeting (2012)
26. Geisel, C.D.: Spacecraft orbit design in the circular restricted three-body problem using higher-dimensional poincaré maps. PhD Dissertation (2013)
27. Caleb, T., Losacco, M., Fossà, A.: Differential algebra methods applied to continuous abacus generation and bifurcation detection: application to periodic families of the Earth-Moon system. *Non-linear Dyn.* **111**, 9721–9740 (2023)
28. Poincaré, H.: *Les Méthodes Nouvelles de la Mécanique Céleste*. Gauthier-Villars et Fils, Paris, France (1892). <https://doi.org/10.1007/BF02742713>
29. Szebehely, V.: *The general and restricted problems of three bodies*. Springer, Vienna (1974). <https://doi.org/10.1007/978-3-7091-2916-6>
30. Moulton, F.: *Periodic Orbits*. Carnegie Institution of Washington, Washington (1920)
31. Howell, K.: Three-dimensional, periodic, ‘halo’ orbits. *Celest. Mech.* **32**, 53–71 (1984). <https://doi.org/10.1007/BF01358403>
32. Howell, K., Campbell, E.T.: Three-dimensional periodic solutions that bifurcate from halo families in the circular restricted three-body problem. *Adv. Astronaut. Sci.* **102**, 891–910 (1999)
33. Sandberg, A., Sands, T.: The Lagrange interpolation polynomial for neural network learning. *Int. J. Comput. Sci. Netw. Secur.* **11**, 255–261 (2011)
34. Higham, N.J.: *Accuracy and stability of numerical algorithms*. SIAM, Philadelphia, USA (1996). <https://doi.org/10.1137/1.9780898718027>
35. Coderre, K., Edwards, C., Cichanet, T. et al.: Concept of operations for the Gateway. In: Pasquier, H., Cruzen, C., Schmidhuber, M., Lee, Y. (eds) *Space operations: inspiring humankind’s future*, pp. 63–82. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11536-4_4
36. Haapala, A.F.: *Trajectory design in the spatial circular restricted three-body problem exploiting three-body problem exploiting higher-dimensional poincare maps*. Purdue University (2014)
37. Canales, D.: *Transfer design methodology between neighborhoods of planetary moons in the circular restricted three-body problem*. Purdue University (2021)
38. Lv, M., Tan, M., Zhou, D.: Design of two-impulse Earth-Moon transfers using differential correction approach. *Aerosp. Sci. Technol.* **60**, 183–192 (2017). <https://doi.org/10.1016/j.ast.2016.11.008>

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.