Analyzing Methods for Unstable Halo Orbit Maintenance

Word count: 4204

***Abstract*—Gravitational systems involving two massive bodies in circular orbits—such as the Sun and Earth, or Earth and the Moon—influencing the motion of a third object of negligible mass, such as a satellite, are ubiquitous in spaceflight. The dynamics of these systems are intricate, leading to the emergence of a number of orbit classes. A halo orbit is a type of periodic, nonplanar orbit existing in such systems that presents substantial utility for a variety of space missions, for instance currently by the James Webb Space Telescope, and in the future by the lunar Gateway station. However, these orbits are inherently unstable, and must be corrected regularly via spacecraft engines. Optimizing this process could present potential for fuel savings, especially due to the exponential relation between velocity change potential and fuel budget. A simulation was created from scratch based on the mathematical model of the described scenario, known as the circular restricted three-body problem. This project uses this simulation to investigate methods for halo orbit "stationkeeping," as such orbital guidance is called. Potential algorithmic improvements for spacecraft set to inhabit halo orbits may offer opportunities for space programs to achieve mission objectives with exponentially smaller fuel budgets for little to no hardware changes, permitting additional payload mass and/or lesser mission costs.**

I. Introduction

A halo orbit is a type of three-body periodic orbit that presents substantial utility for a variety of space program missions, including, but not limited to, the orbit of the James Webb Space Telescope (JWST), launched in 2021 [1], and the planned orbit for the future lunar Gateway station [2]. Such orbits, however, are inherently unstable [1], and require periodic "stationkeeping" thruster burns to maintain. For instance, the JWST, whose Sun–Earth halo orbit has a period of six months, must make orbital correction burns every three weeks [1]. Determining when and how best to perform such correction burns is a nontrivial task, requiring an algorithmic implementation. Thus, careful analysis of algorithms for halo orbit stationkeeping is vital to optimizing for efficiency and reliability.

A. *Circular Restricted Three-Body Problem*

This research analyzes halo orbits within gravitational systems involving two massive bodies orbiting each other in circular motion (such as the Earth and Moon) along with a third object of negligible mass (such as a spacecraft), where it is the third object whose motion is of interest. This problem is known as the **circular restricted three-body problem (CR3BP)** [3], and is complex enough that chaotic patterns emerge in the motion of the third object. Due to the circular nature of the problem, it is convenient to analyze it in a reference frame centered at the **barycenter** (center of mass) of the two massive bodies (around which both orbit) which rotates along with the orbits of the two massive bodies. This is called a **synodic frame** [4]. In this reference frame, the two massive bodies remain in fixed positions. By convention, these positions are placed along the x-axis, with the xy-plane representing the plane of the massive bodies' circular orbits, and the z-axis in the perpendicular direction [3].

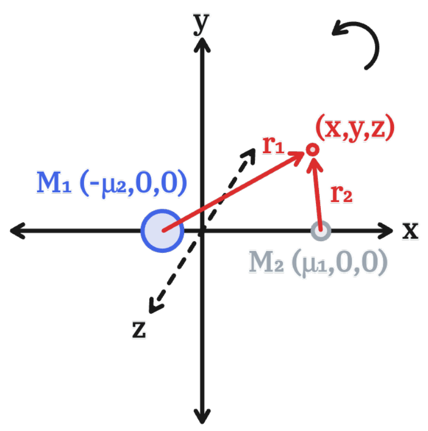

Fig. 1. The CR3BP in a rotating reference frame

For generality, quantities are conventionally standardized to canonical units called distance units (DU), time units (TU), and mass units (MU), such that the distance between the massive bodies is equal to 1 DU, the speed of rotation of the reference frame is 1 radian per TU, and the total mass of both massive bodies is 1 MU [4].

As pictured in Fig. 1, the variable position of the third body is denoted (x, y, z). The distances of the third body from the first and second are respectively denoted r_1 and r_2. The symbols μ_1 and μ_2 respectively represent the masses of the bodies as measured in MU, equal to their mass ratios: $\mu_1 = \frac{M_1}{M_1+M_2}$, and $\mu_2 = \frac{M_2}{M_1+M_2}$.

Note that $\mu_1 + \mu_2 = 1$ and as a result $\mu_2 = 1 - \mu_1$. Positioning the massive bodies at the coordinates shown in the diagram places their barycenter at the origin as desired.

Due to the rotating nature of this reference frame, what are normally "fictitious" forces in **inertial** reference frames, i.e. reference frames that do not accelerate, must be considered: namely, centrifugal force and Coriolis force. As a result, the law of universal gravitation does not directly apply. Instead, the following equations of motion are applicable to the rotating reference frame of the CR3BP [3], [5].

$$\ddot{x} - 2\dot{y} - x = -\frac{\mu_1}{r_1^3}(x + \mu_2) - \frac{\mu_2}{r_2^3}(x - \mu_1)$$

$$\ddot{y} + 2\dot{x} - y = -\frac{\mu_1}{r_1^3}y - \frac{\mu_2}{r_2^3}y$$

$$\ddot{z} = -\frac{\mu_1}{r_1^3} - \frac{\mu_2}{r_2^3}z$$

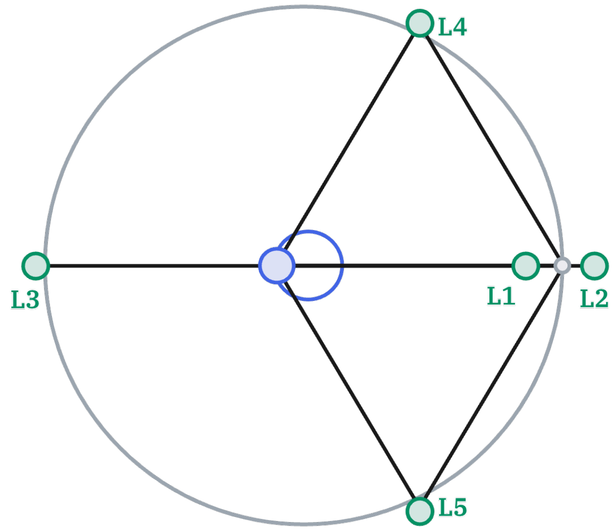

Fig. 2. The five Lagrange points

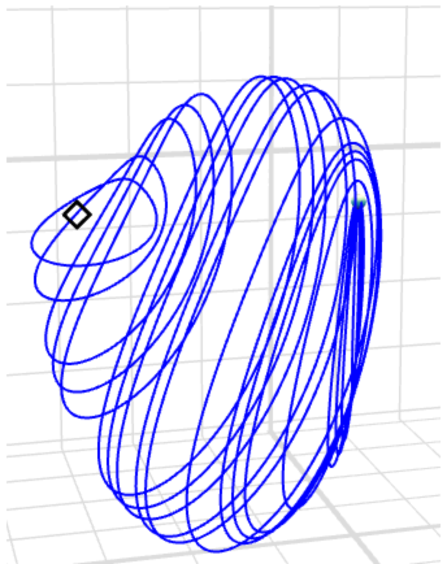

Fig. 3. A family of L2 lunar halo orbits, existing between L2 (left) and the Moon (right) [9]

This system of differential equations does not have a general formulaic solution, and thus must be analyzed approximately via **numerical integration** methods [3].

B. Lagrange Points

While the CR3BP equations of motion do not have a general analytic solution, there do exist five specific solutions where $\ddot{x} = \ddot{y} = \ddot{z} = 0$. These are points of gravitational equilibrium, wherein no net acceleration will be experienced. These points are known as Lagrange points or libration points. If positioned perfectly at one of these points, an object would theoretically remain stationary within the rotating reference frame, maintaining its position relative to the two massive bodies [6], [7].

As shown in Fig. 2, three of the Lagrange points, labelled L1 through L3, lie collinear to the two massive bodies. L1 lies between the bodies, L2 lies on the far side of the second mass, and L3 lies on the far side of the first mass. These are known as the collinear Lagrange points. The other two Lagrange points, L4 and L5, lie at the vertices of equilateral triangles with the line between the two massive bodies as their shared base. L4 lies ahead of the second body in its counterclockwise orbit, and L5 behind [6].

For all practical purposes, it is not possible to position a spacecraft precisely at a Lagrange point of interest. However, due to the interacting gravitational forces of the two massive bodies, it is possible to *orbit* a Lagrange point [8]. One such type of Lagrange point orbit is halo orbits, the subject of this research.

C. Halo Orbits

A **halo orbit** is a type of orbit existing in a circular restricted three-body system which, viewed from the CR3BP rotating reference frame, revolves periodically around any of the three collinear Lagrange points, and maintains its orientation relative to the reference frame [8]. Entire families of such orbits exist for each of the collinear Lagrange points. For instance, halo orbits associated with L2 range from those that closely approach a pole of the smaller body before travelling far away from the opposite pole and in the direction of L2 in an eccentric path—this type called a **near-rectilinear halo orbit (NRHO)**—to orbits that spend all their time farther from both bodies and closer to L2, revolving in a less eccentric path [9], [10]. Each of the loops pictured in Fig. 3 is a different possible southern L2 halo orbit in the Earth–Moon system.

The James Webb Space Telescope was launched in 2021 into a Sun–Earth L2 halo orbit with a period of approximately six months [1]. Orbiting a Lagrange point instead of Earth itself means the Sun and Earth will both always be on the same side of the space telescope, allowing it to keep its heat shield oriented to block heat and light interference from both bodies at all times. Orbiting a Lagrange point instead of occupying a heliocentric (Sun-centered) orbit allows the JWST to maintain proximity to Earth to allow continual communications and data transmission.

The NASA Artemis program plans to establish a lunar station entitled “Gateway” occupying an L2 near-rectilinear halo orbit [2]. This orbit’s close approach to the Moon’s north pole will allow easier access to the lunar surface, while the rotating property of halo orbits means the station will never lose line of sight with Earth due to obstruction by the Moon.

With all the utility of halo orbits, they suffer from the critical fact that they are inherently unstable and, left alone, will decay into a completely different orbit over time [1], [7]. As a result, **stationkeeping** is required via periodic thruster burns in order to maintain any given halo orbit. For instance, the JWST makes orbital corrections on a 21-day cadence [1]. The means of determining when and how to perform stationkeeping burns is a nontrivial task requiring algorithmic implementation.

D. Research Rationale

While algorithms for maintaining halo orbits do already exist —such as that used by the JWST—we do not necessarily know that they are as efficient as they possibly could be. Critically evaluating how the aspects of halo orbit correction affect the resulting costs is important for optimizing efficiency.

Specifically, modifying the parameters of a satellite's halo orbit stationkeeping strategy may decrease the total velocity change (called **delta-v** or Δv) required to maintain the halo orbit for the intended mission duration. According to the ideal rocket equation [11],

$$\Delta v = v_e \ln\left(\frac{m_0}{m_f}\right),$$

the total delta-v of which a spacecraft stage is capable is directly proportional to the fuel ejection velocity v_e, but more importantly, proportional to the logarithm of the ratio of the stage's initial fuelled mass m_0 to its final empty mass m_f. The conclusion to be drawn from this is that achieving a linear decrease in the required delta-v of a mission via a more efficient algorithm for stationkeeping would result in an *exponential* decrease in the required fuel, presenting opportunities for dramatic expenditure reduction should stationkeeping strategies be improved.

II. Methodology

For the purposes of this study, three-body orbital dynamics were studied experimentally using a computer simulation. In light of the fact that it is impractical to attempt to study such dynamics in this way using a real spacecraft, a simulation offers the ability to study the dynamics at the cost of only that required for computation. With a simulation, the experimental design can be based around complete control of the situation and the ability to collect whatever data is necessary about the simulated events, since the entire state of the simulation over time can be reconstructed as necessary.

The simulation itself was conducted via creation of a software using the Rust programming language that uses numerical integration of the aforedescribed CR3BP equations of motion. Specifically, the Runge–Kutta–Fehlberg variable–step size numerical integration method (RKF45) was used due to its capability to adapt simulated step size to the dynamics of the system to satisfy a designated error bound, allowing very high levels of simulation precision to be achieved while remaining within computational limits.

To put the equations of motion into a form for which this integration method can be used, the state of the system must be combined into a single six-dimensional vector consisting of the three position components and the three velocity components. Then, the derivative of this state vector is equal to another six-dimensional vector consisting of the three original velocity components along with the values for the three acceleration components as determined by the CR3BP equations of motion. This creates a first-order vector differential equation that can then be solved numerically using the RKF45 method.

Let v be the aforedescribed state vector of the system, ε be the configurable error bound parameter, and h be the step size that adapts over the course of the numerical integration process to satisfy the error bound ε. The following equations then describe the procedure of RKF45 [12] for the case where the derivative of the state vector does not depend on time, which is true in this scenario since the mathematical model used does not incorporate outside influences. This means that this method approximates future values of v given that $\dot{v} = f(v)$ for some f, as opposed to $\dot{v} = f(t, v)$, the time-dependent case. In this scenario, $f(v)$ performs the operation described before, building the six-dimensional derivative vector using the velocity components from the input combined with the acceleration components computed using the CR3BP equations of motion. The initial state of the system is taken to be v_0, and the subsequent outputs of the numerical integration process are v_i for $i \geq 1$.

$$k_1 = hf(v_{i-1})$$

$$k_2 = hf\left(v_{i-1} + \frac{1}{4}k_1\right)$$

$$k_3 = hf\left(v_{i-1} + \frac{3}{32}k_1 + \frac{9}{32}k_2\right)$$

$$k_4 = hf\left(v_{i-1} + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\right)$$

$$k_5 = hf\left(v_{i-1} + \frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\right)$$

$$k_6 = hf\left(v_{i-1} - \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\right)$$

$$v_i = v_{i-1} + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5$$

$$\tilde{v}_i = v_{i-1} + \frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6$$

$$R = \frac{1}{h}|\tilde{v}_i - v_i|$$

$$\delta = 0.84\left(\frac{\varepsilon}{R}\right)^{\frac{1}{4}}$$

$$\begin{array}{ll} \text{if } R \leq \varepsilon & \text{keep } v_i \text{ as the current step solution} \\ & \text{and move to the next step with step size } \delta h \\ \text{if } R > \varepsilon & \text{recalculate the current step with step size } \delta h \end{array}$$

The simulation, when run, records the system as a series of entries, where each entry includes the time at which it occurred along with the six numbers representing the state of the system—the three position components and the three velocity components. Because the simulation operates within the synodic frame, the massive gravitating bodies do not move and thus no information needs to be recorded about them over time to faithfully reconstruct the simulated system. The only necessary information in that regard is one of their mass ratios, μ_1 or μ_2.

For the purpose of simulating halo orbits, initial conditions for position and velocity were sourced from the Three-Body Periodic Orbits database provided by the Jet Propulsion Laboratory [9]. This database provides initial conditions and other parameters for a number of CR3BP orbits of different varieties, existing in different two–gravitational body systems, including Sun–Earth, Earth–Moon, Jupiter–Europa, and others. The database possesses an abundant selection of halo orbits, including those revolving around all three collinear

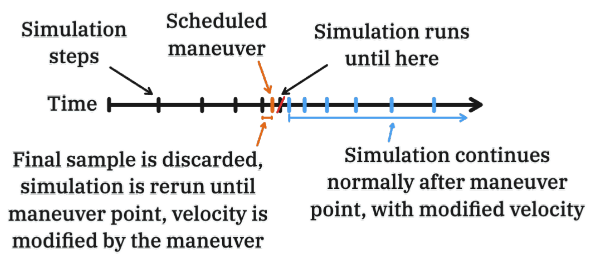
Simulation steps
Scheduled maneuver
Simulation runs until here
Time
Final sample is discarded, simulation is rerun until maneuver point, velocity is modified by the maneuver
Simulation continues normally after maneuver point, with modified velocity

Fig. 4. The mechanism for simulating maneuvers while maintaining precision

Lagrange points. This study only considered halo orbits in the Earth–Moon system.

For the experiment, the CR3BP simulation was modified to allow **impulsive** orbital corrections or **maneuver**s. To simulate a scheduled maneuver without losing precision, the simulation runs until it just passes the time of the maneuver, then deletes the final sample that came after the scheduled maneuver time, ensuring the time interval between the prior sample and the maneuver itself is smaller than necessary to maintain the precision goal. Then, the simulation is rerun until the exact time of the maneuver, and the resulting sample has its velocity modified by the correction vector of the maneuver, and the simulation continues as normal after that point. This process is visualized in Fig. 4.

Then, a broad stationkeeping algorithm similar to that used by the James Webb Space Telescope was implemented to interface with the rest of the simulation by producing such maneuvers, and the simulation was subsequently used to model orbits such as L2 southern halo orbit 77 in the database, shown in Fig. 5. The algorithm functions as follows [1].

Halo orbits in the CR3BP are symmetrical about the xz-plane, so it is known that a spacecraft travelling in an optimal halo orbit will always cross the xz-plane once every half-period, and will cross it perpendicularly, indicating that at every half-period, the spacecraft is expected to satisfy the constraints $y = 0$, $\dot{x} = 0$, and $\dot{z} = 0$ as closely as possible, and any deviation from these ideals at each half-period can be considered erroneous.

This corrective algorithm employs a linear estimation of the accumulation of error over particular time intervals using a construct called a **state-transition matrix (STM)**. This matrix is computed by numerically integrating a particular matrix differential equation with an integration interval equal to the interval to model error accumulation over [1], [7]. At a point in time at which a corrective maneuver is to be created, the future propagation of the simulation without corrections is considered up until the fourth expected crossing of the xz-plane after the time of the maneuver [13]. The same RKF45 numerical integration method from before is used to compute the state-transition matrix from the planned time of the maneuver to the expected time of said fourth crossing. The error at this crossing point is known to be equal to any deviation from zero in the y, x-velocity, or z-velocity components of the state vector, and the approximation being used relates error at the end of the time interval δ_f to error at

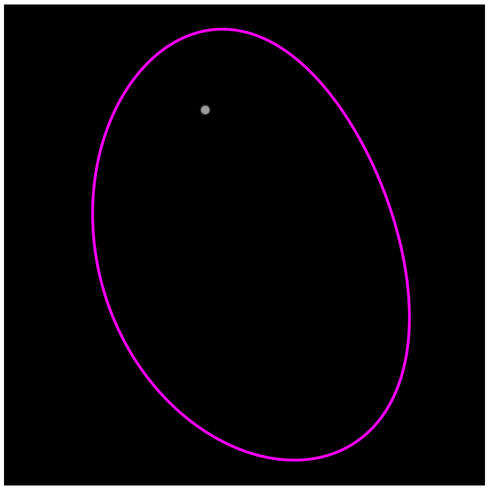

Fig. 5. L2 southern halo orbit 77

the start of the time interval δ_0 with the STM Φ by $\Phi\delta_0 = \delta_f$. However, consider that only three of the six state components are constrained and thus have a known error value, and that only three of the six state components—those corresponding to velocity—can be directly adjusted by corrections—directly modifying position would correspond to teleportation, which is of course impossible. In this manner, there are only three reasonable input components to the matrix product and three expected output components from it. As such, a relevant 3×3 sub-matrix of the STM can be taken by keeping only the rows corresponding to constrained variables with known errors and keeping only the columns corresponding to the velocity components of the state vector.

With this component removal, δ_0 is now a 3-vector equal to the velocity error at the maneuver time, and δ_f is a 3-vector equal to the y-position, x-velocity, and z-velocity error at the anticipated time of the fourth xz-plane crossing from the maneuver time. Since δ_f and Φ are both known, the velocity error at the maneuver time can be found by $\delta_0 = \Phi^{-1}\delta_f$, and an appropriate correctional velocity change would simply be a negation of this error.

However, it is known that non-complex eigenvalues of the STM, which appear in one or more reciprocal pairs, correspond to pairs of stable–unstable directions in state space [1]. Perturbations along an eigenvector corresponding to a real eigenvalue with magnitude less than 1 are corrected on their own by the dynamics of the system, while if the eigenvalue has magnitude greater than 1, perturbations compound exponentially [10]. A general error vector would have components in both of these directions, but presumably, for the sake of efficiency, error in stable directions would not need to be manually corrected. As such, in the final stationkeeping implementation, instead of finding the correction based on the error directly, the error is first projected via a dot product onto the eigenvector of the STM corresponding to the real eigenvalue of the greatest magnitude, which is the dominant unstable direction. The maneuver is then scheduled to make a correction to only this unstable component, leaving the stable component to correct itself.

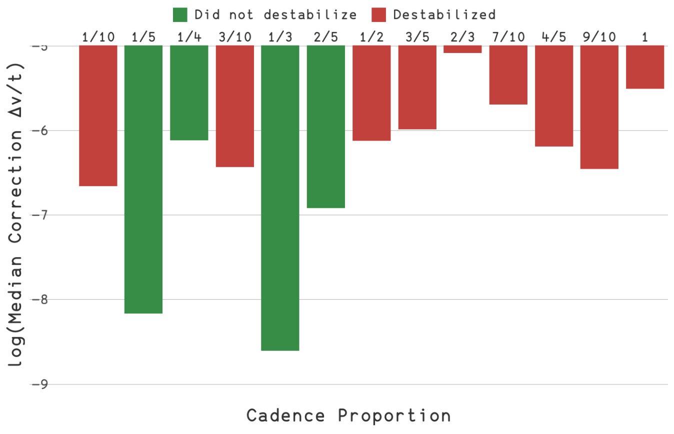

Fig. 6. Logarithmic plot of correction cadence delta-v

In addition to writing files containing logs of the evolution of the orbital simulation, the simulation was made to log all instances of orbital correction burns, including the times at which they occurred and the velocity changes made, allowing the complete reconstruction of any simulation run afterward.

Several different correction cadences, specified as fractions of orbital period, were tested with halo orbits from the Three-Body Periodic Orbits database. Cadences of $\frac{1}{10}, \frac{1}{5}, \frac{1}{4}, \frac{3}{10}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{7}{10}, \frac{4}{5}, \frac{9}{10}$, and 1 times the orbital period were tested. If the orbit remained stable for 4000 time units, it was recorded as "never" destabilizing and the median delta-v cost per time unit was recorded. If the orbit destabilized within 4000 time units, the time of destabilization was recorded and the median delta-v cost per time unit was recorded up until the point of destabilization.

Since, in real life, spacecraft engines are not perfectly reliable, and our instruments do not have unlimited precision and will not allow correction maneuvers to be executed perfectly, the simulation was modified to add small randomized perturbations when the spacecraft conducts calculated velocity changes. Perturbations are determined by proportional alterations in the magnitude of executed correction maneuvers, as governed by a configurable perturbation strength multiplier. Stationkeeping cost over 1000 time units was then evaluated for several different perturbation strength levels.

III. Results & Discussion

Since the stationkeeping algorithm tested does not specify exactly when/how often correctional maneuvers should be performed, part of what was tested was the effect of using different cadences of regular maneuvers on the achieved stability level and the resulting delta-v expenditure. For the purpose of standardizing across trials of different durations, stationkeeping costs are measured as median delta-v expenditure per time unit, computed by dividing each maneuver's velocity change magnitude by the amount of time between that maneuver and the next, then taking the median of all results for maneuvers performed while the orbit in question maintained stability. Fig. 6 shows the results of this element of experimentation.

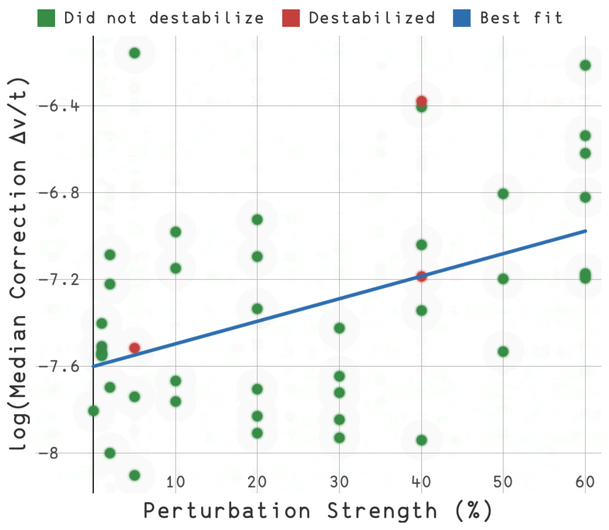

Fig. 7. Logarithmic plot of delta-v cost against perturbation strength (correlation coefficient 0.440)

One result of interest is that of the correction cadences tested, those equal to a quotient of orbital period with a small odd number, particularly 3 and 5, seem to correlate with orbital stability and lower stationkeeping costs. Cadences that are not simple fractions of period seemingly lead to worse orbital stability. However, this pattern shows a notable exception at $\frac{1}{2}$. One possibility for why this is the case is that half-period orbital corrections can only ever occur at the crossings of the xz-plane. However, recall that the correction system relies on constraining what velocities are possible at the xz-plane. As a consequence, it is possible that half-period corrections are less reliable because the system is not able to make effective corrections without contradicting its own constraints. Another notable aspect is the significant differences in efficiency resulting from small changes in correction cadence. For instance, $\frac{3}{10}$ and $\frac{1}{3}$ are numerically similar (respectively 0.3 and $0.\overline{3}$), but a $\frac{3}{10}$ correction cadence led to 100 times higher costs and an overall lack of stability compared to $\frac{1}{3}$.

In addition to testing different orbital correction cadences, the stationkeeping scheme was tested under the influence of small randomized perturbations to the executed maneuvers. In real life, no spacecraft can perfectly execute a particular velocity change; there will always be some amount of error resulting from physical systems. To simulate this, perturbations were made randomly to the magnitude of each maneuver executed, determined proportionally to the overall size of the maneuver. Data relating the strength of these perturbations to the resulting delta-v expenditure was collected.

The points pictured in Fig. 7 represent simulations of the same halo orbit from before, using consistent quarter-period correction cadence, under the influence of different strengths of perturbations. Maximal perturbation strength as a percentage of maneuver magnitude is shown along the x-axis, and median correction cost is plotted logarithmically along the y-axis. Green dots represent simulated orbits that remained stable for 1000 time units, and the shown correction cost is averaged over this entire time interval. Red dots represent simulated orbits that destabilized at some point before 1000

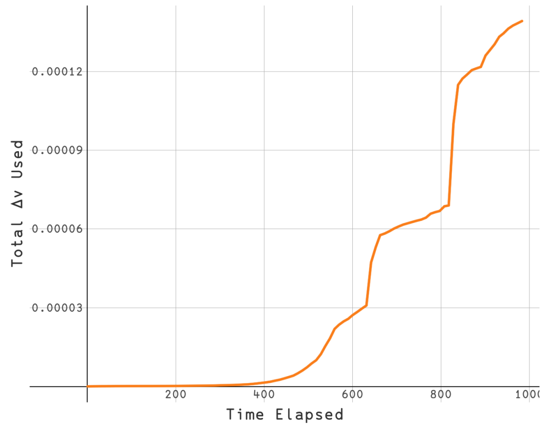

Fig. 8. Cumulative delta-v expenditure over time

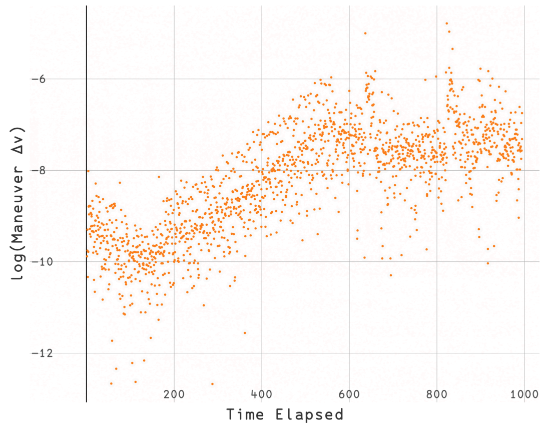

Fig. 9. Logarithmic plot of the delta-v expenditure of individual maneuvers

time units had elapsed, and their portrayed correction cost data is averaged only until the point of destabilization. The line of best fit is based only on the data from orbits that remained stable. The data shown seems to imply that there may be a linear relationship between the logarithm of correction cost and the proportional strength of perturbations, indicating an exponential relation with the correction cost itself. However, the correlation coefficient of only 0.440 means that this data is ultimately inconclusive on this claim. Recall that the stochastic perturbations represent imprecision in a spacecraft's systems' ability to make precise velocity changes. In other words, if there does exist a true linear relationship as exhibited by this graph, it would imply an exponential relation between spacecraft maneuver execution precision and stationkeeping cost.

Qualitatively, it was observed that when cumulative expended delta-v was graphed as a function of time, it seemed to unpredictably alternate between exhibiting linear-reminiscent growth and exhibiting exponential-reminiscent growth, often switching abruptly. Pictured in Fig. 8 is one instance of the aforementioned cumulative expended delta-v graph. The x-axis represents time elapsed in time units, and the y-axis represents total delta-v expended until that point in distance units per time unit. There is a clear sharp turn around 550 time units elapsed, where the growth profile switches from apparently exponential to apparently linear. The slope then changes a number of times after that. It is unclear what the cause of this peculiar behavior is, as throughout this process, the orbit itself remains essentially unchanged. Another simulation of this orbit even switched between linear and exponential growth modes four times.

As a general trend, it was observed that, as time passes, the average size of orbital correction maneuvers often tends to increase exponentially or remain approximately constant. Fig. 9 shows a logarithmic scatter plot of the velocity changes made by each individual maneuver in the same simulation instance as in Fig. 8. Since the plot is logarithmic, the clear linear growth between approximately 160 and 550 TU demonstrates exponential growth of the overall maneuver size. The overall trend then becomes roughly flat, although changing height abruptly, reflecting the sudden changes in slope in the previous graph. Another curious feature of this plot is the linear decrease from 0 to about 160 TU.

IV. Conclusions

In the context of the linearized unstable error reversal stationkeeping method implemented based on the algorithm used by the JWST, results seem to indicate that small changes in the frequency of orbital corrections can result in dramatic differences in stability and delta-v cost. This implies that if a space program seeks to station a satellite in a halo orbit, it could stand to benefit significantly from careful consideration for the frequency with which corrections are performed. Particularly, greater stability and significantly lower costs were observed when conducting corrections at a cadence equal to a small-denominator fraction of orbital period that is smaller than half of the period.

This research gave some indication, but was ultimately inconclusive on whether there exists a truly exponential relationship between maneuver execution imprecision and stationkeeping delta-v costs. If this is a true exponential relation, there could exist potential for significant delta-v cost reductions through improving precision of instruments used to execute orbital corrections.

V. Limitations & Future Studies

The data collected revealed perplexing oscillatory trends between modes of growth pertaining to delta-v expenditure, presenting an avenue of investigation for further research on this topic, which could potentially lead to an even deeper understanding of how stationkeeping cost evolves.

It should be noted that while the CR3BP is a good approximation of many physical systems, it still neglects aspects that could non-negligibly impact results—no orbit is perfectly circular and no gravitational system completely isolated—so it would be beneficial to repeat this study using an ephemeris model, which uses real celestial body motion data to account for non-circular planetary motion and external influences.

Additionally, this study focused primarily on a particular overall algorithm based on that used by the JWST. However, this does not take into consideration alternative algorithms that work entirely differently, many of which have been proposed by different publications. Future research could benefit from studying these alternative stationkeeping algorithms, since they may end up being more efficient in general.

Glossary

barycenter
: the center of mass of two or more gravitating bodies

circular restricted three-body problem (CR3BP)
: a simplified gravitational system model consisting of two bodies of considerable mass circularly orbiting their mutual barycenter, and a third object of negligible mass (such as a satellite) whose motion is studied

delta-v
: (also called Δv) a measure of a spacecraft's capability for velocity change; often used as a more versatile measure of fuel budget, since the amount of change a satellite can make to its orbital velocity directly affects what maneuvers it can perform

halo orbit
: a type of three-body periodic orbit revolving around any one of the three collinear Lagrange points

impulsive
: (of a velocity change) considered to be effectively instantaneous

inertial
: (of a reference frame) not accelerating

Lagrange points
: five points of gravitational equilibrium as viewed in the rotating reference frame of the circular restricted three-body problem

libration points
: synonymous with Lagrange points

maneuver
: a planned adjustment made to a satellite's orbit, usually through the use of onboard thrusters to modify the satellite's velocity

near-rectilinear halo orbit (NRHO)
: a subtype of halo orbits that make a close, fast approach to one pole of the smaller orbited body before travelling far away from the opposite pole and in the direction of the orbited Lagrange point, creating a highly elongated shape that is considered to be almost a line, hence "near-rectilinear"

numerical integration
: the process of computationally approximating the integral of a complicated function, used to estimate solutions to differential equations without analytic solutions

state-transition matrix (STM)
: a 6×6 matrix used to approximate how error in the six-dimensional state vector of a halo orbit changes between two particular points in time

stationkeeping
: the process of conducting orbital correction maneuvers in order to keep a satellite in an orbit which would otherwise destabilize

synodic frame
: a non-inertial frame of reference with its origin at the barycenter of two circularly orbiting bodies which rotates in sync with the orbit of the two bodies, such that the bodies remain aligned along the x-axis of the frame

References

[1] J. Petersen, "L2 Station Keeping Maneuver Strategy for the James Webb Space Telescope," NASA, Aug. 2019. [Online]. Available: https://ntrs.nasa.gov/citations/20190028877

[2] S. Fuller, E. Lehnhardt, C. Zaid, and K. Halloran, "Gateway program status and overview," *Journal of Space Safety Engineering*, vol. 9, no. 4, pp. 625–628, Dec. 2022, doi: 10.1016/j.jsse.2022.07.008.

[3] B. Weber, "Circular Restricted Three-Body Problem." [Online]. Available: https://orbital-mechanics.space/the-n-body-problem/circular-restricted-three-body-problem.html

[4] W. Koon, M. Lo, J. Marsden, and S. Ross, *Dynamical Systems, the Three-Body Problem and Space Mission Design*. California Institute of Technology, 2006. [Online]. Available: https://www.cds.caltech.edu/~marsden/volume/missiondesign/KoLoMaRo_DMissionBk.pdf

[5] G. Miceli, N. Bosanac, and R. Karimi, "Generating the Trajectory Design Space for Neptunian System Exploration," Sep. 2024, [Online]. Available: https://www.colorado.edu/faculty/bosanac/sites/default/files/2024-09/MicBosKar_2024ASC_NeptuneTrajectories.pdf

[6] B. Weber, "Application of the CR3BP: Lagrange Points." [Online]. Available: https://orbital-mechanics.space/the-n-body-problem/lagrange-points.html

[7] F. Vega, Z. Manchester, M. Lo, and R. Restrepo, "Contingency-aware station-keeping control of halo orbits," 2024, doi: 10.48550/ARXIV.2405.20458.

[8] in *Basics of Spaceflight*, NASA Science, 2023. [Online]. Available: https://science.nasa.gov/learn/basics-of-space-flight/chapter5-1

[9] Jet Propulsion Laboratory, "Three-Body Periodic Orbits." [Online]. Available: https://ssd.jpl.nasa.gov/tools/periodic_orbits.html

[10] D. Davis *et al.*, "Orbit Maintenance and Navigation of Human Spacecraft at Cislunar Near Rectilinear Halo Orbits," San Antonio, Texas: NASA, Feb. 2017. [Online]. Available: https://ntrs.nasa.gov/citations/20170001347

[11] K. Tsiolkovsky, *Reactive Flying Machines*. 1954.

[12] Y. Wang, "Runge–Kutta Method." [Online]. Available: https://math.okstate.edu/people/yqwang/teaching/math4513_fall11/Notes/rungekutta.pdf

[13] L. Bucci, M. Lavagna, and R. Jehn, "Station Keeping Techniques for Near Rectilinear Halo Orbits in the Earth-Moon system," Salzburg, Austria, May 2017. [Online]. Available: https://www.researchgate.net/publication/317427621

Appendix: Excerpt from Rust code written to model the circular restricted three-body problem

```rust
/// Dimensionless circular restricted three-body system
#[derive(Clone)]
pub struct Cr3bs {
    // μ₂ = m₂ / (m₁ + m₂)
    pub mass_ratio: f64,
    pub period: f64,
    pub maneuvers: Vec<Maneuver>,
    pub time_log: Vec<f64>,
    pub pos_log: Vec<Vec3>,
    pub vel_log: Vec<Vec3>,
    pub jacobi_log: Vec<f64>,
    pub orbit_report: Option<OrbitReport>,
}

impl Cr3bs {
    // Create a new circular restricted three-body system instance
    pub fn new(mass_ratio: f64, pos0: Vec3, vel0: Vec3) -> Self {
        let mut new = Self {
            mass_ratio,
            period: f64::INFINITY,
            maneuvers: Vec::new(),
            time_log: vec![0.0],
            pos_log: vec![pos0],
            vel_log: vec![vel0],
            jacobi_log: Vec::new(),
            orbit_report: None,
        };
        let init_jacobi = new.jacobi(pos0, vel0);
        new.jacobi_log.push(init_jacobi);
        new
    }

    // Inform the instance of the expected orbital period
    pub fn with_period(mut self, period: f64) -> Self {
        self.period = period;
        self
    }

    // Vector from mass 1 to the satellite
    fn r1(&self, pos: Vec3) -> Vec3 {
        Vec3::x() * self.mass_ratio + pos
    }

    // Vector from mass 2 to the satellite
    fn r2(&self, pos: Vec3) -> Vec3 {
        Vec3::x() * (self.mass_ratio - 1.0) + pos
    }

    // Compute acceleration using the CR3BP equations of motion
    fn calc_accel(&self, pos: Vec3, vel: Vec3) -> Vec3 {
        let mu2 = self.mass_ratio;
        let mu1 = 1.0 - mu2;

        let r1p3 = self.r1(pos).norm().powi(3);
        let r2p3 = self.r2(pos).norm().powi(3);

        Vec3::new(
            pos.x + 2.0 * vel.y - mu1 * (pos.x + mu2) / r1p3 - mu2 * (pos.x - mu1) / r2p3,
            pos.y - 2.0 * vel.x - mu1 * pos.y / r1p3 - mu2 * pos.y / r2p3,
            -mu1 * pos.z / r1p3 - mu2 * pos.z / r2p3,
        )
    }

    // Interpolate to find the approximate position at a particular time
    pub fn pos_at(&self, t: f64) -> Option<Vec3> {
        let j = self.time_log.partition_point(|&cur_t| cur_t <= t);
        if j == self.time_log.len() {
            return None;
        }
        let i = j - 1;
        let lerp_factor = (t - self.time_log[i]) / (self.time_log[j] - self.time_log[i]);
        Some(self.pos_log[i] + lerp_factor * (self.pos_log[j] - self.pos_log[i]))
    }

    // Interpolate to find the approximate velocity at a particular time
    pub fn vel_at(&self, t: f64) -> Option<Vec3> {
        let j = self.time_log.partition_point(|&cur_t| cur_t <= t);
        if j == self.time_log.len() {
            return None;
        }
        let i = j - 1;
        let lerp_factor = (t - self.time_log[i]) / (self.time_log[j] - self.time_log[i]);
        Some(self.vel_log[i] + lerp_factor * (self.vel_log[j] - self.vel_log[i]))
    }

    // Interpolate to find the approximate state vector at a particular time
    pub fn state_at(&self, t: f64) -> Option<Vec6> {
        self.pos_at(t)
            .and_then(|p| self.vel_at(t).map(|v| mk_state(p, v)))
    }

    // Jacobian matrix, used to find the state-transition matrix
    fn jacobian(&self, t: f64) -> Option<Mat6> {
        let pos = self.pos_at(t)?;

        let mu2 = self.mass_ratio;
        let mu1 = 1.0 - mu2;

        let r1p3r = r1(self.mass_ratio, pos).norm().powi(3).recip();
        let r1p5r = r1(self.mass_ratio, pos).norm().powi(5).recip();
        let r2p3r = r2(self.mass_ratio, pos).norm().powi(3).recip();
        let r2p5r = r2(self.mass_ratio, pos).norm().powi(5).recip();

        let [x, y, z] = <[f64; 3]>::from(pos);
        let xd1 = x - mu1;
        let xd2 = x + mu2;

        // Hessian partial derivatives
        let xx = 1.0 - mu1 * (r1p3r - 3.0 * xd2 * xd2 * r1p5r) - mu2 * (r2p3r - 3.0 * xd1 * xd1 * r2p5r);
        let xy = mu1 * 3.0 * y * xd2 * r1p5r + mu2 * 3.0 * y * xd1 * r2p5r;
        let xz = mu1 * 3.0 * z * xd2 * r1p5r + mu2 * 3.0 * z * xd1 * r2p5r;
        let yy = 1.0 - mu1 * (r1p3r - 3.0 * y * y * r1p5r) - mu2 * (r2p3r - 3.0 * y * y * r2p5r);
        let yz = mu1 * 3.0 * y * z * r1p5r + mu2 * 3.0 * y * z * r2p5r;
        let zz = -mu1 * (r1p3r - 3.0 * z * z * r1p5r) - mu2 * (r2p3r - 3.0 * z * z * r2p5r);

        Some(Mat6::from_rows(
            &[
                [0.0, 0.0, 0.0, 1.0, 0.0, 0.0],
                [0.0, 0.0, 0.0, 0.0, 1.0, 0.0],
                [0.0, 0.0, 0.0, 0.0, 0.0, 1.0],
                [xx, xy, xz, 0.0, 2.0, 0.0],
                [xy, yy, yz, -2.0, 0.0, 0.0],
                [xz, yz, zz, 0.0, 0.0, 0.0],
            ]
            .map(From::from),
        ))
    }

    /// State-transition matrix
    pub fn stm(&self, t0: f64, t1: f64, dt0: f64, epsilon: f64) -> Option<Mat6> {
        if t1 <= t0 {
            return None;
        }
        let mut t = t0;
        let mut state = Mat6::identity();
        let mut dt = dt0;

        while t < t1 {
            dt = dt.min(t1 - t);

            let jacobian = self.jacobian(t)?;
            let (ord4, ord5) = calc_rkf45_step(dt, state, |s| jacobian * s);
            let error = (ord4 - ord5).norm_squared() / 6.0;
            if error <= epsilon {
                state = ord4;
                t += dt;
            }
            if error.is_nan() {
                println!("STM computation failed");
                dbg!(ord4);
                dbg!(ord5);
            }
            adjust_rkf45_dt(&mut dt, epsilon, error);
        }

        Some(state)
    }
}
```