A Bi-impulsive Transfer Trajectory Between the Earth and Moon in CR3BP

a project presented to

The Faculty of the Department of Aerospace Engineering

San José State University

In partial fulfillment of the requirement for the degree

Master of Science in Aerospace Engineering

by

Jay Mehta

December 2021

approved by

Prof. Jeanine Hunter

Faculty Advisor

ABSTRACT

A Bi-impulsive Transfer Trajectory Between the Earth and Moon

Jay Mehta

A Bi-impulsive transfer between Earth and Moon is examined in this paper. The trajectory simulation is done initially in MATLAB and then compared it with a NASA GMAT simulation. Initial conditions for the orbit are taken from a previously published data and validity of the MATLAB code is tested by comparing the simulation result with the previously published data and the GMAT simulation data. In GMAT the trajectory transfer is solved by using a B-Plane transfer and certain design decisions are made to make GMAT simulation as close to a CR3BP.

ACKNOWLEDGEMENTS

I would like to thank my family and friends for their continuous support throughout my bachelor's and master's degree. A special thanks to both of my advisor, Dr. Capdevila for helping me start an orbital mechanic's project and professor Hunter, who helped me fine tune the project once Dr. Capdevila had to take leave of absence. Thank you to Takoua Bejaoui (TK) for helping me understand B-Plane transfer. Also, thanks to E.M. Leonardi master student at Sapienza University of Rome for answering my questions via email and helping me understand his published paper.

TABLE OF CONTENTS

List of Figures

List of Tables

Symbols

Symbol	Definition	Unit (SI)
ΔV	Delta Velocity	$\frac{km}{s}$
δ	Phase Angle	deg
μ	Gravitational Constant	$\frac{km^3}{\text{sec}^2}$
m	Mass	kg
LEO	Low Earth Orbit	$----$
LMO	Low Moon Orbit	$----$
B-Plane	Body plane	$----$
ToF	Time of Flight	$days$
CR3BP	Circular Restricted 3 Body Problem	$----$
Ω	Angular Velocity	$\frac{rad}{sec}$
$\boldsymbol{r}$	Position vector	km
v	Velocity vector	$\frac{km}{s}$
$\boldsymbol{a}$	Acceleration vector	$\frac{km}{\text{sec}^2}$

1. Introduction

NASA plans to send humans back on the Moon by 2024 [1]. This sparked an increased interest in the Lunar exploration missions. In order to send humans and robotic missions to the Moon efficiently, different optimal low and/or high thrust trajectory transfers are being studied. The most simple and fast but not energy efficient approach is the Hohmann transfer[2]. Hohmann transfer requires two burns, one at the perigee of the orbit and another at the apogee. Spacecraft are placed at perigee while in Earth parking orbit and apogee is set at the desired Moon's orbit altitude. Another way to examine transfer of a spacecraft from Earth to Moon is by using the patched-conics method. The patched-conics approximation relies on the Keplerian decomposition of the solar system dynamics [3]. By carefully switching the SOI (Sphere of Influence) along the orbit, the spacecraft's motion is only governed by one primary body at a given time. For example, in the case of Earth to Moon transfer using patched conics, spacecraft will be in Earth's SOI for most of the transfer and by only the Moon during the final time. Both the Hohmann transfer, and patched conics are the simple, direct method of transfer in 2BP (2 body problem). Some alteration of Hohmann and patched conics transfer was used in all lunar missions from the 1960s to the 1980s, including the Luna and Apollo mission. The 2BP transfer to the Moon is limited by the launch window and requires multiple corrections burns, increasing the total ΔV cost. In the case of Apollo 11, it had to perform two Lunar orbital Intersection burns and four midcourse corrections. The total ΔV required for Apollo 11 to be in orbit around the Moon was 13571.1 ft/s (4.136 km/s) [4].

below summarizes all the burn estimation to be performed by Apollo 11 Command Service Module (CSM) to get to the Low Moon Orbit (LMO).

Table 1.1- CSM burn schedule [4]

BURN / MANEUVER	GETI BURN TIME ΔVC	ATTITUDE (DEG) LH/LV INERTIAL	LIGHTING	ΔV (FPS)	ULLAGE	TVC MODE	REFSMMAT
S-IVB TLI	02:44:26 5 MIN 20 SEC		BURNOUT AT SUNRISE	ΔVX: -- ΔVY: -- ΔVZ: -- ΔV REQ: 10,451.2	----	----	PAD
CSM/LM S-IVB EVASIVE MNVR	04:39:44.9 2.8 SEC 15.6 FPS		DAYLIGHT	ΔVX: 5.1 ΔVY: 0.0 ΔVZ: 19.0 ΔV REQ: 19.7	NOT REQUIRED	G&N AUTO	PAD
MIDCOURSE CORRECTIONS MCC_1 TO MCC_4	11:45 26:45 53:55 70:55		----	ΔVX: NOMINALLY ΔVY: ZERO ΔVZ: ΔV REQ:	NOT REQUIRED	G&N AUTO	PAD PTC PTC LDG SITE
LOI_1	75:54:28.4 5 MIN 58.9 SEC 2914.8 FPS		DAYLIGHT (SS -1 HR 7 MIN)	ΔVX: -2891.8 ΔVY: -433.1 ΔVZ: 20.4 ΔV REQ: 2924.1	NOT REQUIRED	G&N AUTO	LDG SITE
LOI_2	80:09:29.7 16.4 SEC		DAYLIGHT (SR +9 MIN)	ΔVX: 138.3 ΔVY: 0.0 ΔVZ: 75.9 ΔV REQ: 157.8	2 JET 20 SEC	G&N AUTO	LDG SITE
CSM/LM SEP	100:39:50.4 8 SEC		SUNLIGHT (SS -14 MIN)	ΔVX: 0.0 ΔVY: 0.0 ΔVZ: 2.5 ΔV REQ: 2.5	----	G&N AUTO	LDG SITE
*CSM PLANE CHANGE	107:05:33.4 0.8 SEC 5.7 FPS		DARKNESS (SS +17 MIN)	ΔVX: 0.0 ΔVY: 16.6 ΔVZ: 0.0 ΔV REQ: 16.6	2 JET 20 SEC	G&N AUTO	PLANE CHANGE
LM JETTISON	131:53:04.7 3.1 SEC 0.8 FPS		DAYLIGHT (SR +36 MIN)	ΔVX: -1.0 ΔVY: -- ΔVZ: -- ΔV REQ: 1.0	----	G&N AUTO	LIFT OFF
TEI	135:24:33.8 2 MIN 29.4 SEC NOT AVAILABLE		DAYLIGHT (SR +10 MIN)	ΔVX: 3213.3 ΔVY: 705.0 ΔVZ: -138.8 ΔV REQ: 3292.7	2 JET 16 SEC	G&N AUTO	LIFT OFF
MIDCOURSE CORRECTIONS MCC_5 TO MCC_7	150:24 172:00 192:06		----	ΔVX: ΔVY: NOMINALLY ΔVZ: ZERO ΔV REQ:	----	G&N AUTO	PTC PTC ENTRY

A trajectory calculated using CR3BP, a circular restricted three-body problem, can provide a more accurate trajectory. As the name suggests, the perturbation caused by the third body, for this project, Moon, is accounted in the motion of the spacecraft. A detailed derivation of Equation of Motion (EOM) for an Earth-Moon, Circular Restricted Three-Body Problem is developed in the later section of this paper. There are still assumptions made in a CR3BP, such as both Earth and Moon are orbiting each other in a circular orbit, but the trajectory designed using CR3BP provides accurate ΔV approximation over a 2BP (Two body problem).

1.1 Problem Definition

To design a Bi-impulsive transfer trajectory from Low Earth Orbit (LEO) to Low Moon Orbit (LMO) in CR3BP in MATLAB and simulate it also in NASA's GMAT (General Mission Analysis Tool). LEO altitude of 463 km and LMO altitude of 100 km is selected for the mission. Identical altitudes are also used in NASA GMAT. ODE45 function is used to integrate the EOMs in MATLAB.

2. Literature Review

Two body orbital mechanics have been shaped by Kepler's laws combined with Newton's law of motion. Two-body orbital mechanics have been studied and researched extensively and are also the foundation for 3-body and N-body problems.

2.1 CR3BP orbit transfer between Earth and Moon

Belbruno carried out research for low energy transfer between Earth and Moon using a weak stability boundary. In his paper, he numerically demonstrated that the ΔV needed for the transfer was 18% less than that of the Hohmann transfer. The time of flight (TOF) for such a transfer was upwards of 3-5 months, not ideal for a manned mission but suitable for a robotic mission. Belbruno's weak stability ballistic boundary transfer has been successfully demonstrated by the Japanese spacecraft Hiten, which arrived at Moon on October 2, 1991 [5]. Belbruno's work laid the foundation for the ballistic transfer between Earth and Moon.

Bi-Impulsive transfer in the CR3BP is also studied extensively. The goal of the study has always been to find the minimal ΔV require for a permanent lunar capture. The study conducted by Qi and Xu, students at Beijing universities, compared the ΔV for Lunar transfer using the patched conics method and in CR3BP [6]. Table 2.1 below provides a summary of their result for ΔV needed for patched conics transfer vs. transfer done in CR3BP.

Table 2.1 - Quick summary of ΔV needed for patched conics transfer vs. CR3BP [6]

Semi Major Axis around Moon [km]		ΔV [km/s]
Patched Conics	23,300	3.453796
CR3BP	23,663	3.099713
Patched Conics	25,066	3.552857
CR3BP	25,268	3.103431

Qi and Xu also numerically optimized their CR3BP model to get an optimal Lunar transfer with minimum ΔV. For the minimum theoretical ΔV they got a transfer time of infinite, so such a transfer is not realistically possible. Figure 2.1 below summarizes the integral Jacobi Constant for permanent Lunar orbit vs. the minimum ΔV needed for such a transfer.

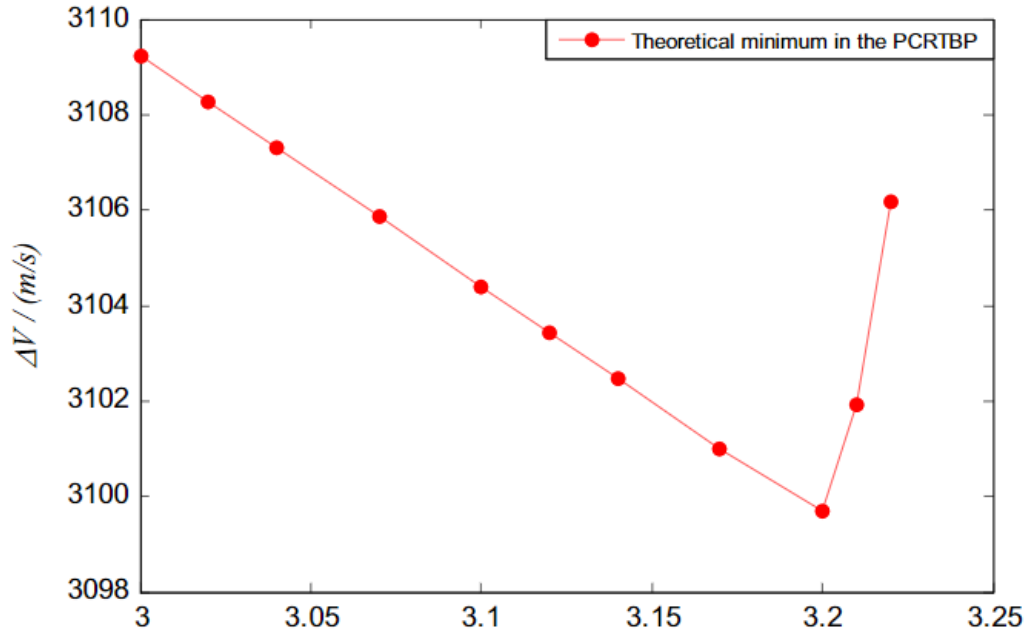


Figure 2.1 - Jacobi constant vs ΔV [6]

Giovani Mengali and Alessandro Quarta, two researchers from the University of Pisa, also performed optimization of Bi-impulsive Trajectory in CR3BP for Earth-Moon transfer by constraining TOF (Time of Flight) [7]. Spacecraft was parked in a circular orbit of radius 6545 km around the Earth and transferred to a circular orbit around Moon of radius1840 km. They took the mission data from Belbruno and Miller [5] and used it as an initial condition for their WSB (weak stability boundaries) approach in the CR3BP environment to validate their 3-body model. They found the almost equivalent ΔV and transfer time was obtained without using the sun perturbation. Table 2.2 below summarizes their result.

Table 2.2. - Summary of performance of different optimal transfer from earth circular orbit to moon circular orbit [7]

Transfer Type	ΔV [km/s]	TOF (days)
Weak Stability Boundary	3.838	140
Biparabollic	3.953	Infinite
Hohmann (2body Problem)	3.991	5
Bielliptic	4.148	90

The optimal transfer done by Mengali and Quarta is ideal for robotics mission but are not suitable for human flights due to their long transfer time. Optimization Study done by Miele and Mancuso accounts for flight time suitable for a human mission, thus providing a bit more realistic trajectories for the human Moon mission. They set up their problem in a simplified restricted three-body model and optimized it using the sequential gradient-restoration algorithm. The parameter to be optimized were the initial phase angle of the spacecraft with respect to Earth and Moon, flight time, and the velocity impulse at departure and the arrival. Phase Angle also referred as departure angle, is defined as the angle between v_0 of the spacecraft and the inertial

X-axis of the Earth [Figure 2.2]. Table 2.3 below summarizes their Earth-Moon transfer results. The critical thing to note is the transfer time, 4.5 days.

Table 2.3 - Earth to moon flight, clockwise LMO arrival [8]

Altitude LMO (km)	ΔV_{total} (km/s)	ΔV_{Earth} (km/s)	ΔV_{Moon} (km/s)	Phase Angle δ (deg)	TOF (days)
100	3.882	3.068	0.814	-116.88	4.50
200	3.868	3.068	0.800	-116.88	4.50
300	3.855	3.068	0.787	-116.88	4.50

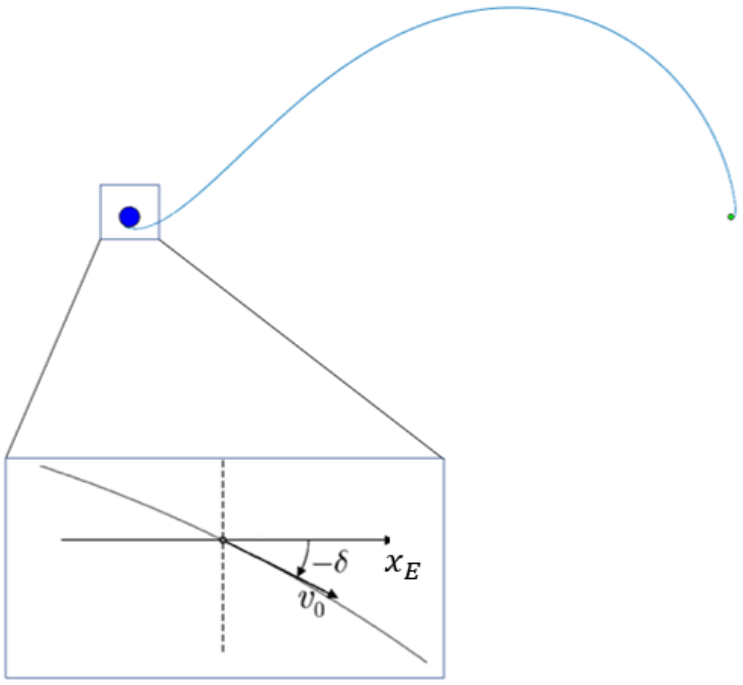


Figure 2.2 - Definition of phase angle δ

Building on Miele and Mancuso's research, Leonardi and Potani also perform a similar study in two- and Three-dimensional Earth-Moon orbit transfer [9]. For their 2-D CR3BP (Planer CR3BP) model, they use an LEO altitude of 463 km and LMO altitude of 100 km, similar to the initial conditions used by Miele. Unlike Miele, their results are slightly more accurate because they did not identify the center of Earth as the center of the entire system.

Table **2.4** below summarizes their optimized two-dimensional LEO-LMO orbit transfer for clockwise arrival at Moon. Their paper does not explicitly state the time of flight, but it's a reasonable assumption to assume it is similar to TOF found by Miele and Mancuso. The trajectory solved using MATLAB in this paper uses the initial condition provided by reference 9.

Table 2.4 - Globally optimal two-dimensional LEO-LMO orbit transfer [9]

J- Jacobi Constant(km/s)	ΔV_{LEO}	δ deg
3.885	3.069	-117.52

2.2 NASA's GMAT

NASA GMAT, General Mission Analysis Tool, is an enterprise, multi-mission opensource software system for space mission design, optimization, and Navigation [10]. It was developed by NASA engineers, private industry, and other public and private contributors. It has been used in real-world missions, most notably, Lunar Reconnaissance Orbiter (LRO), Transiting Exoplanet Survey Satellite (TESS), OSIRIS-Rex. Figure 2.3 below lists the NASA mission whose trajectory was designed using the GMAT. Some key features of GMAT are,

- Dynamic and Environmental Modeling: GMAT contains high fidelity dynamics models including gravity, drag, tides, Solar Radiation Perturbation (SRP). It also includes information on the constellation, high fidelity ephemerides data, and a rich set of coordinate systems such as J2000, ICRF, body fixed, body rotating, and many others.
- Plotting, Reporting, and Product Generation: GMAT has tools like interactive 3-D graphics, customizable plots, and reports, post computational animation, which can be used to analyze data outside GMAT.
- Optimization and Targeting: GMAT can be used to solve boundary value targets, nonlinear problems, and constrained optimization problems.
- Programming Infrastructure: GMAT uses MATLAB syntax to define equations in its script. Users can define any variable, array, or string and solve it using the GMAT. It also has MATLAB and Python interfaces and can easily use those tools.
- Orbit Determination Infrastructure: GMAT has a Batch estimator, extended Kalman Filter smoother, error modeling, process noise modeling, and many more incorporated to determine the ideal orbit and analyze the result.

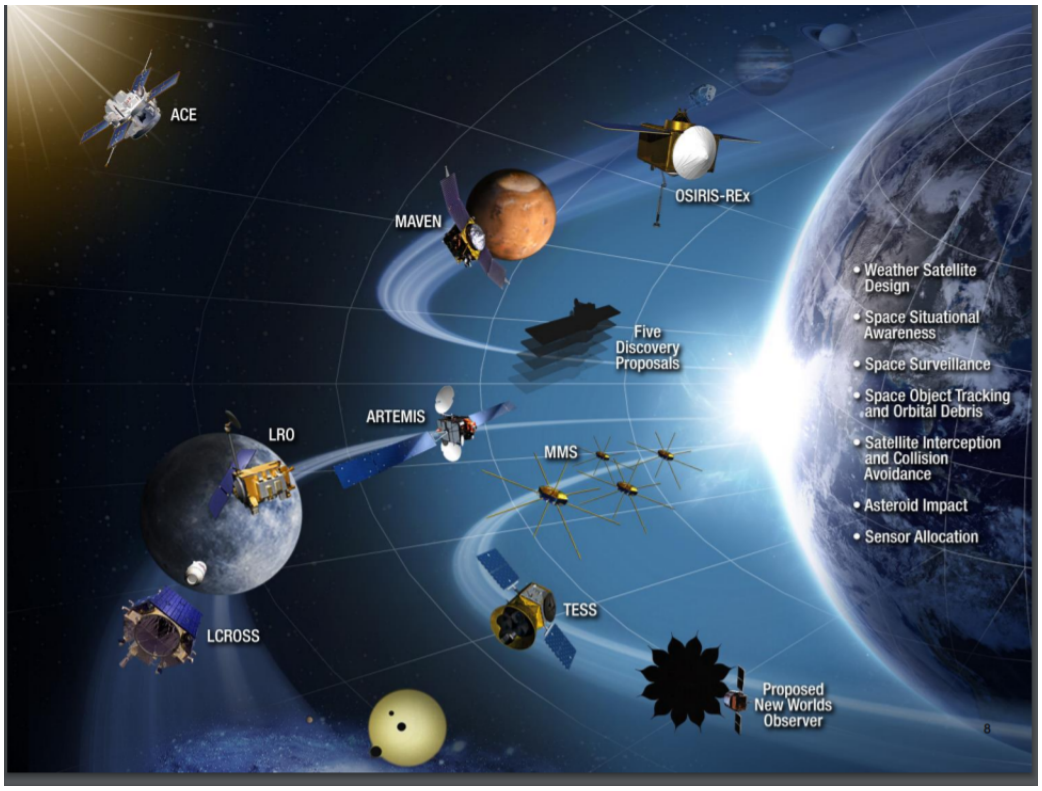


Figure 2.3 - List of NASA mission whose trajectory was optimized using GMAT [11]

2.3 B-Plane

The B-Plane transfer is used in this project to transfer spacecraft from LEO to LMO in NASA GMAT. B-plane, also called a body plane, is an imaginary plane that contains the target body (Moon) and is orthogonal to the incoming asymptote of the spacecraft. A spacecraft approaching the target body is assumed to be in a hyperbolic orbit. In this case, the target insertion point is behind the target body, and the relationship between the target insertion point and the current velocity is nonlinear due to the gravitational attraction of the target body. Therefore, to adjust the miss distance caused by the perturbation forces, B-Plane targeting is adopted [11]. B-Plane targeting allows for a linear relationship between the target B-Vector and the instantaneous velocity of the spacecraft [12]. A simple B-Plane calculation is performed in the lesson learned section of this paper.

Figure 2.4 below illustrates the B-Plane and its vector. To describe the vectors on the B-plane, unit vectors $\hat{R}$ and $\hat{T}$ $a$re used.

$$\hat{T} = \frac{\hat{S} \times \hat{N}}{||\hat{S} \times \hat{N}||} \qquad 2.1$$

Where, $\hat{S}$ is a unit vector parallel to the spacecraft excess velocity from the C.G. (center of gravity) of the target body and $\hat{N}$ is a normal unit vector of the target planet's equatorial plane. $\hat{R}$ vector can be defined using the right-hand rule, i.e.

$$\widehat{\mathrm{R}} = \hat{S} \times \widehat{\mathrm{T}} \qquad 2.2$$

By using these vectors, B-Vector can be defined as follows

$$\widehat{\mathrm{B}} = B_T\widehat{\mathrm{T}} \times \mathrm{B_R}\widehat{\mathrm{R}} \qquad 2.2$$

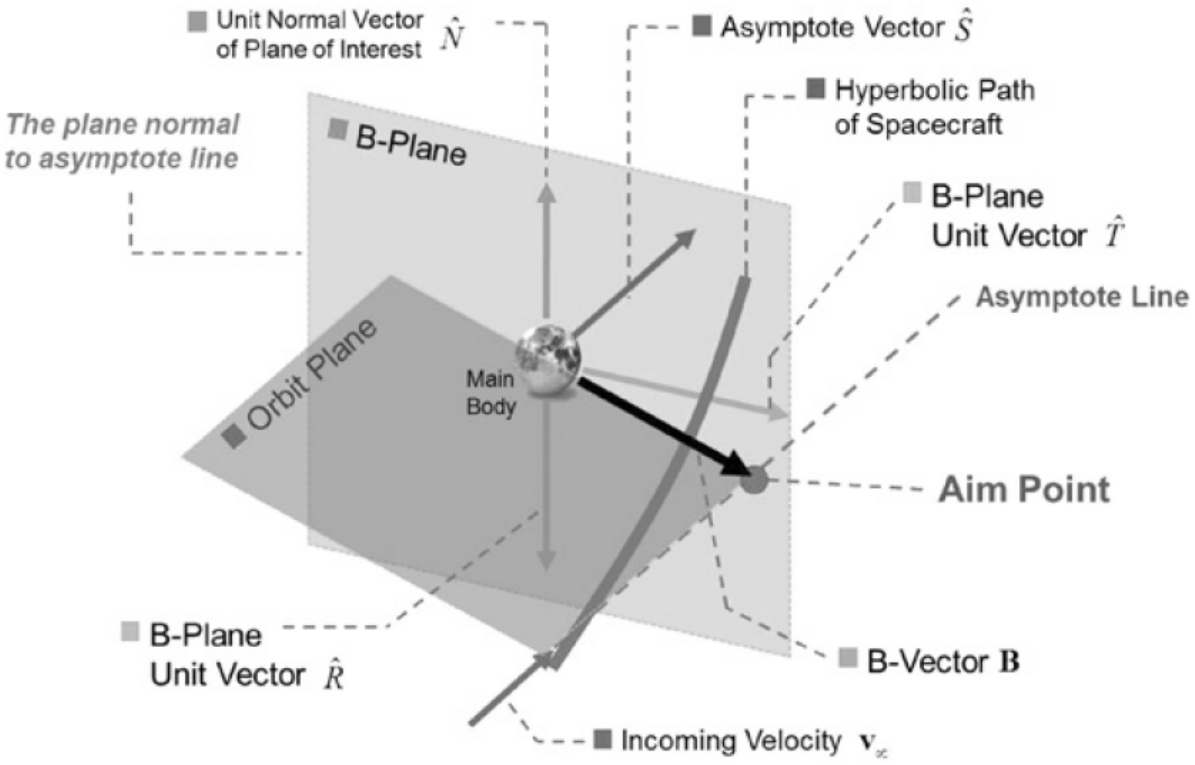


Figure 2.4 - B-plane definition

The complete derivation of B-Plane can be found in the appendix of reference 13. GMAT adopted Kizner works, and the following two conditions need to be satisfied to set up a B-plane project in GMAT [13]. Figure 2.5 below shows the geometry of the B-Plane as seen from the viewpoint perpendicular to the B-plane.

- GMAT should know r and v in F_1coordinate (GMAT default coordinate frame)
- F_B is the coordinate system GMAT will perform the B-plane calculation in. GMAT will place the T in the XY-plane of F_B and F_B should have a gravitational body at its origin. If GMAT fails to find the gravitational body at its origin, it will error.

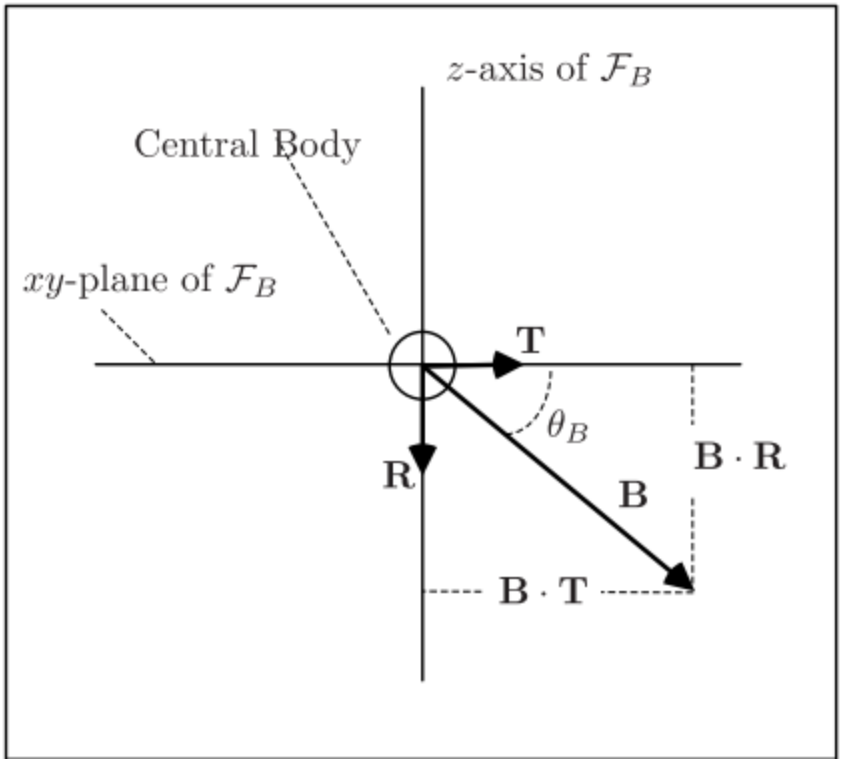


Figure 2.5 - Geometry of the B-Plane as seen from a viewpoint perpendicular to the B-plane

3. Methodology

This section discusses the methodology used to simulate the orbital trajectory using MATLAB and GMAT, starting with getting the equation of motion in CR3BP.

3.1 Circular Restricted 3-body Problem Equation of Motion

This section derives the equation of motion in CR3BP for Earth and Moon. Consider the mass of Earth as m_E and mass of Moon as m_M. The constant scalar distance from the Earth to Moon is r_{EM} as shown in Figure 3.1. Let's also consider a non-Newtonian reference frame at the barycenter of Earth and Moon, B_0, such that the x-axis, $\widehat{\boldsymbol{b}}_{\boldsymbol{x}}$, is directed towards Moon, as shown Figure 3.1. The Y-axis, $\widehat{\boldsymbol{b}}_{\boldsymbol{y}}$, lies in the orbital plane to which the z-axis,$\widehat{\boldsymbol{b}}_{\boldsymbol{z}}$, is perpendicular. The angular velocity of B around the barycenter as seen from a Newtonian reference frame N (Sun reference frame) is given by,

$$^{N}\boldsymbol{\Omega}^{B} = \Omega\, \widehat{\boldsymbol{b}_{z}} \tag{3.1}$$

Where,

$$\Omega = \frac{2\pi}{T} \tag{3.2}$$

And T is the period of the orbit,

$$T = \frac{2\pi}{\sqrt{\mu}} \times r_{EM}^{\frac{3}{2}} \tag{3.3}$$

Therefore,

$$\Omega = \sqrt{\frac{\mu}{r_{EM}^{3}}} \tag{3.4}$$

Where,

$$\mu = GM = G(m_E + m_M) \tag{3.5}$$

For a planer CR3BP, Earth and Moon lie in the orbit plane; hence their y and z coordinates are zero. To determine their location on the x-axis, let's use the center of mass equation.

$$m_E x_E + m_M x_M = 0 \tag{3.6}$$

Since r_{EM} is known,

$$x_M = x_E + r_{EM} \tag{3.7}$$

From equation 9 and 10

$$x_E = -\frac{m_M}{m_E + m_M} r_{EM} \tag{3.8}$$

And

$$x_M = \frac{m_E}{m_M + m_E} r_{EM} \tag{3.9}$$

Now to fully set up a Three-Body problem, let's introduce the third body, i.e., the spacecraft of mass m_S. Since the $m_s, \ll m_E, m_M$, spacecraft mass does not affect the motion of the primary body, but the Earth and Moon govern spacecraft motion. Unlike the 2-Body problem, spacecraft motion in the 3-Body problem has no general, closed-form solution. By setting up the Equation of Motion (EOM) and integrating it over time, spacecraft trajectory in the three-body problem can be calculated.

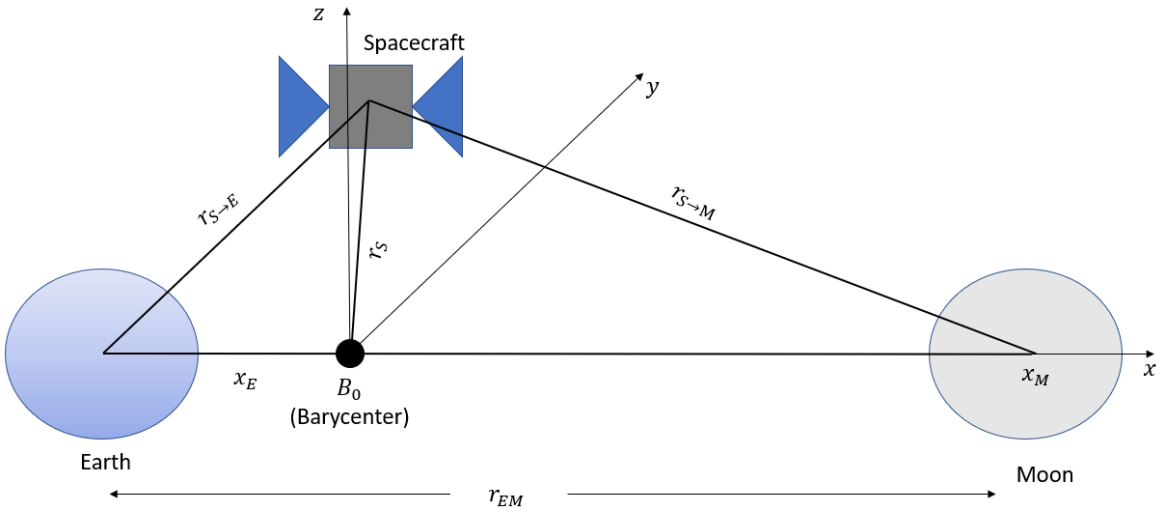


Figure 3.1- Earth-moon 3 body set up

In a barycenter frame, the position of the spacecraft w.r.t the Earth is given by,

$$^{E}\vec{\mathbf{r}}^{S} = (x_S - x_E)\,\hat{b}_x + y_S\,\hat{b}_y + z_S\,\hat{b}_z \tag{3.10}$$

$$= \left(x_S + \frac{m_M}{m_E + m_M} r_{EM}\right)\hat{b}_x + y_S\hat{b}_y + z_S\hat{b}_z \tag{3.11}$$

Spacecraft position w.r.t. Moon,

$$^{M}\vec{\mathbf{r}}^{S} = \left(x_S - \frac{m_E}{m_E + m_M} r_{EM}\right)\hat{b}_x + y_S\,\hat{b}_y + z_S\,\hat{b}_z \tag{3.12}$$

And the position of the spacecraft w.r.t. the barycenter

$$^{B}\vec{\mathbf{r}}^{S} = x_s\,\hat{b}_x + y_s\,\hat{b}_y + z_S\,\hat{b}_z \tag{3.13}$$

The inertial velocity of the spacecraft can be found by taking the time derivative of equation 3.13 w.r.t the Newtonian reference frame N_o, from the center of Sun.

$$ {}^{N}\boldsymbol{v}^{S} = \frac{{}^{B}d}{dt}\,{}^{B}\vec{\mathbf{r}}^{S} + {}^{N}\boldsymbol{\Omega}^{B} \times {}^{B}\vec{\mathbf{r}}^{S} = {}^{B}\vec{\mathbf{v}}^{S} + {}^{N}\boldsymbol{\Omega}^{B} \times {}^{B}\vec{\mathbf{r}}^{S} \quad (3.14) $$

and,

$$ {}^{B}\vec{v}^{S} = Spacecraft\ velocity\ w.r.t\ the\ barycentre\ = \dot{x}_s\hat{b}_x + \dot{y}_s\hat{b}_y + \dot{z}_s\hat{b}_z \quad (3.15) $$

And the acceleration of the spacecraft can be calculated by taking the time derivative of equation 3.14.

$$ {}^{N}\vec{\mathbf{a}}^{S} = \frac{{}^{B}d}{dt}\,{}^{B}\vec{\mathbf{v}}^{S} + {}^{N}\boldsymbol{\Omega}^{B} \times {}^{B}\vec{\mathbf{v}}^{S} \quad (3.16) $$

$$ = {}^{B}\vec{\mathbf{a}}^{S} + \dot{\boldsymbol{\Omega}} \times {}^{B}\vec{\mathbf{r}}^{S} + \boldsymbol{\Omega} \times \left(\boldsymbol{\Omega} \times {}^{B}\vec{\mathbf{r}}^{S}\right) + 2\boldsymbol{\Omega} \times {}^{B}\vec{\mathbf{v}}^{S} $$

The rotational velocity of the barycenter is constant, thus $\dot{\boldsymbol{\Omega}}=\mathbf{0}$

$$ {}^{N}\vec{\mathbf{a}}^{S} = \boldsymbol{\Omega} \times \left(\boldsymbol{\Omega} \times {}^{B}\vec{\mathbf{r}}^{S}\right) + 2\boldsymbol{\Omega} \times {}^{B}\vec{\mathbf{v}}^{S} + {}^{B}\vec{\mathbf{a}}^{S} \quad (3.17) $$

$$ {}^{B}\vec{\mathbf{a}}^{S} = \ddot{x}_S\hat{b}_x + \ddot{y}_S\hat{b}_x + \ddot{z}_S\hat{b}_x \quad (3.18) $$

Substituting equation 3.14,3.15 in 3.16,

$$ {}^{N}\vec{\mathbf{a}}^{S} = \Omega\hat{b}_z \times \left[\Omega\hat{b}_z \times \left(x_s\hat{b}_x + y_s\hat{b}_y + z_s\hat{b}_z\right)\right] + 2\left(\Omega\hat{b}_z\right) \times \left(\dot{x}_s\hat{b}_x + \dot{y}_s\hat{b}_y + \dot{z}_s\hat{b}_z\right) + \ddot{x}_S\hat{b}_x + \ddot{y}_S\hat{b}_x + \ddot{z}_S\hat{b}_x \quad (3.19) $$

$$ {}^{N}\vec{\mathbf{a}}^{S} = -\Omega^2 \times (x_s\hat{b}_x + y_s\hat{b}_y) + 2\Omega\,\dot{x}\hat{b}_x - 2\Omega\dot{y}\hat{b}_x + \ddot{x}_S\hat{b}_x + \ddot{y}_S\hat{b}_y + \ddot{z}_S\hat{b}_z \quad (3.20) $$

Reorganizing all the terms

$$ {}^{N}\vec{\mathbf{a}}^{S} = (\ddot{x} - 2\Omega\dot{y} - \Omega^2 x)\hat{b}_x + (\ddot{y} + 2\Omega\dot{x} - \Omega^2 y)\hat{b}_y + \ddot{z}\hat{b}_z \quad (3.21) $$

The force acting on the satellites are gravitational forces caused by the Earth and Moon. Therefore, by Newton's second law

$$ F = ma = F_E + F_M \quad (3.22) $$

$$ F_E = G * \frac{m_E m_S}{r_{S\to E}^3}\boldsymbol{r}_{\boldsymbol{S\to E}} \quad (3.23) $$

$$ F_M = G * \frac{m_M m_S}{r_{S\to M}^3}\boldsymbol{r}_{\boldsymbol{S\to M}} \quad (3.33) $$

Let,

$$ \mu_E = Gm_E\ and\ \mu_M = Gm_M \quad (3.34) $$

$$ {}^{N}\vec{\mathbf{a}}^{S} = (\ddot{x} - 2\Omega\dot{y} - \Omega^2 x)\hat{b}_x + (\ddot{y} + 2\Omega\dot{x} - \Omega^2 y)\hat{b}_y + \ddot{z}\hat{b}_z = -\frac{\mu_E}{r_{S\to E}^3}\left[\left(x_S + \frac{m_M}{m_E + m_M}r_{EM}\right)\hat{b}_x + y_S\hat{b}_y + z_S\hat{b}_z\right] - \frac{\mu_M}{r_{S\to M}}\left(x_S - \frac{m_E}{m_E + m_M}r_{EM}\right)\hat{b}_x + y_S\hat{b}_y + z_S\hat{b}_z \quad (3.35) $$

Taking dot product of equation 3.35 w.r.t $\hat{b}_x$

$$(\ddot{x} - 2\Omega\dot{y} - \Omega^2 x) = -\frac{\mu_E}{r_{S\to E}^3}\left(x_S + \frac{m_M}{m_E + m_M} r_{EM}\right) - \frac{\mu_M}{r_{S\to M}}\left(x_S - \frac{m_E}{m_E + m_M} r_{EM}\right) \quad 3.36$$

Taking dot product of equation 3.35 w.r.t $\hat{b}_y$

$$(\ddot{y} + 2\Omega\dot{x} - \Omega^2 y) = -\frac{\mu_E}{r_{S\to E}^3} y_S - \frac{\mu_M}{r_{S\to M}} y_S \quad 3.37$$

Taking dot product of equation 3.35 w.r.t $\hat{b}_z$

$$-\frac{\mu_E}{r_{S\to E}^3} z_S - \frac{\mu_M}{r_{S\to M}} z_S \quad 3.38$$

The equation of 3.36, 3.37, and 3.38 will be integrated using ODE45 to plot the spacecraft's trajectory under the gravitational force of Earth and Moon.

3.2 Trajectory simulation using MATLAB

Initially, the orbit transfer is numerically solved using MATLAB and its native ODE45 function. To identify the successful LEO-LMO transfer in two dimensions, having the proper initial condition is essential. For this project, the optimal conditions needed for the two-dimensional transfer are taken from the paper published by Leonardi and Pontani [9]. The optimal transfer velocity, ΔV and the phase angle δ calculated by Leonardi in his paper are extremely close to the values found by Miele and Mancuso [8]. **Error! Reference source not found.** below illustrates the relationship between the ΔV and δ calculated by Leonardi. In plot the J=$\Delta V_{LEO} + \Delta V_{LMO}$. Table 3.1 summaries the ΔV_{LEO} and δ phase angle used for this project.

Table 3.1 - Initial condition used for MATLAB trajectory

	ΔV_{LEO} [km/s]	δ[deg]
Clockwise	3.069	-117.52

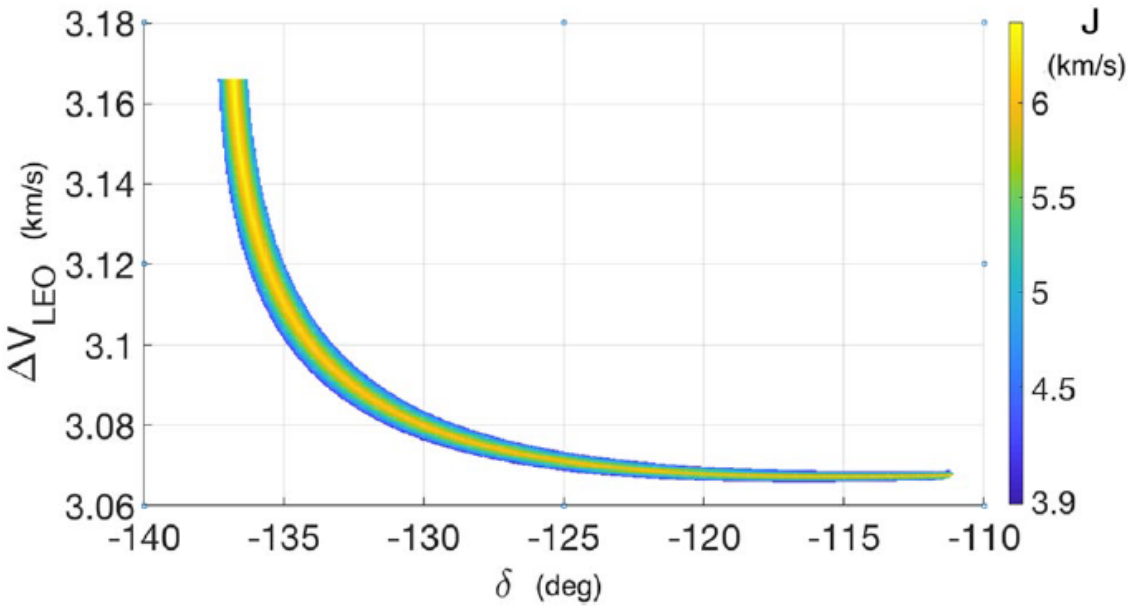


Figure 3.2 - Contour plot of the objective function [9]

Following assumptions are also made to investigate the LEO-LMO transfer.

- The Spacecraft trajectory lies entirely on the Moon orbital plane, i.e., it's a planar Circular Restricted Three-Body Problem.
- The third body perturbation in LEO and LMO is neglected.
- Two impulsive burns will complete the transfer, the initial one at LEO called ΔV_{LEO} and the second one at LMO to circularize the orbit called ΔV_{LMO}
- Spacecraft attitude at LEO is selected at 453 km,
- Spacecraft altitude at LMO is chosen at 100 km.
- EOM is numerically solved using the engineering units (km, km/s, kg) and not in canonical forms (D.U., T.U.).

The position of the spacecraft where the tangential ΔV_{LEO} burn takes place is calculated using the following equation.

$$x_S = x_E + R_{LEO}cos\delta \tag{3.39}$$

$$y_S = R_{LEO}sin\delta \tag{3.40}$$

Where,

$$R_{LEO} = R_{Earth} + SpaceCraft\ Altitude\ at\ LEO \tag{3.41}$$

$$\dot{x}_S = (\Omega R_{LEO} - v_0) \times sin\delta \tag{3.42}$$

$$\dot{y}_S = (v_0 - \Omega R_{LEO})\cos\delta \tag{3.43}$$

Where

$$v_0 = \sqrt{\frac{\mu_E}{R_{LEO}}} + \Delta v_{LEO} \tag{3.44}$$

The initial velocity of the spacecraft at LEO is calculated using equation 3.44 Table 3.2 below summarizes other constants used to solve the trajectory using MATLAB numerically.

Table 3.2 - Initial condition used in MATLAB

Constant	Value	Units
Gravitational Constant	6.67×10^{-20}	$\frac{km^3}{kg\,\sec^2}$
Radius of Earth	6378	km
Radius of Moon	1737	km
Distance between Earth and Moon	384400	km
Mass of Earth	5.9724×10^{24}	kg
Mass of Moon	0.07346×10^{24}	kg
μ_{Earth}	398600	$\frac{km^3}{\sec^2}$
μ_{Moon}	4903.02	$\frac{km^3}{\sec^2}$

The EOM of motion listed in equations 28.1 and 28.2 is integrated using ODE45 until the orbital parameter y_{final}=-1837 km is achieved. Once the desired altitude around Moon is achieved, the spacecraft's velocity w.r.t barycenter is converted to the velocity of spacecraft w.r.t. Moon. The angle θ between the $\boldsymbol{r}_{s\to m}$ and the x-axis is given by,

$$sin\,\theta = \frac{y_{final}}{R_{LMO}} \tag{3.45}$$

$$cos\theta = \frac{x_{final} - x_{Moon}}{R_{LMO}} \tag{3.46}$$

Where, $x_{final} and\ y_{FInal}$ are final spacecraft position numerically integrated using ODE45. The velocity of the spacecraft required for the circular orbit at LMO is calculated using the equation listed below,

$$\dot{x}_{LMO} = \sqrt{\frac{\mu_{moon}}{R_{LMO}}}\sin\theta + \Omega y_{\text{final}} \tag{3.47}$$

$$\dot{y}_{LMO} = -\sqrt{\frac{\mu_{Moon}}{R_{LMO}}}cos\theta - \Omega(x_{final} - x_{MOON}) \tag{3.48}$$

Thus,

$$\Delta V_{LMO} = \sqrt{(\dot{x}_{LMO} - \dot{x}_{final})^2 + (\dot{y}_{LMO} - \dot{y}_{final})^2} \tag{3.49}$$

3.3 GMAT Simulation

NASA GMAT is also used to simulate similar transfer orbit. Spacecraft is parked in the circular orbit around the Earth at an altitude of 463 km, and the GMAT problem is set up such that it uses B-Plane transfer to reach 100 km altitude above Moon. Later another problem is set up in GMAT to achieve a circular orbit at 100 km altitude around Moon. GMAT uses Rung-

Kutta 89 solver to solve the boundary value problem. This section shows users how the problem is set up in GMAT, design decisions made while selecting specific settings, and a brief explanation of certain GMAT features.

The first step in GMAT is to set up the spacecraft's circular parking orbit around the Earth. To define the orbit around the Earth, a modified Keplerian state type is selected. The radius of perigee and radius of apogee is inputted as 6840 km, and the inclination of the orbit is kept at 25°. RAAN, AOP and TA are kept at 0°. Moon orbit around the Earth has an inclination of 18.28°-28.58° [14]; thus, providing an initial inclination of 25° to the satellite is kept inplane with Earth-Moon. Figure 3.3 below summarizes the initial spacecraft property.

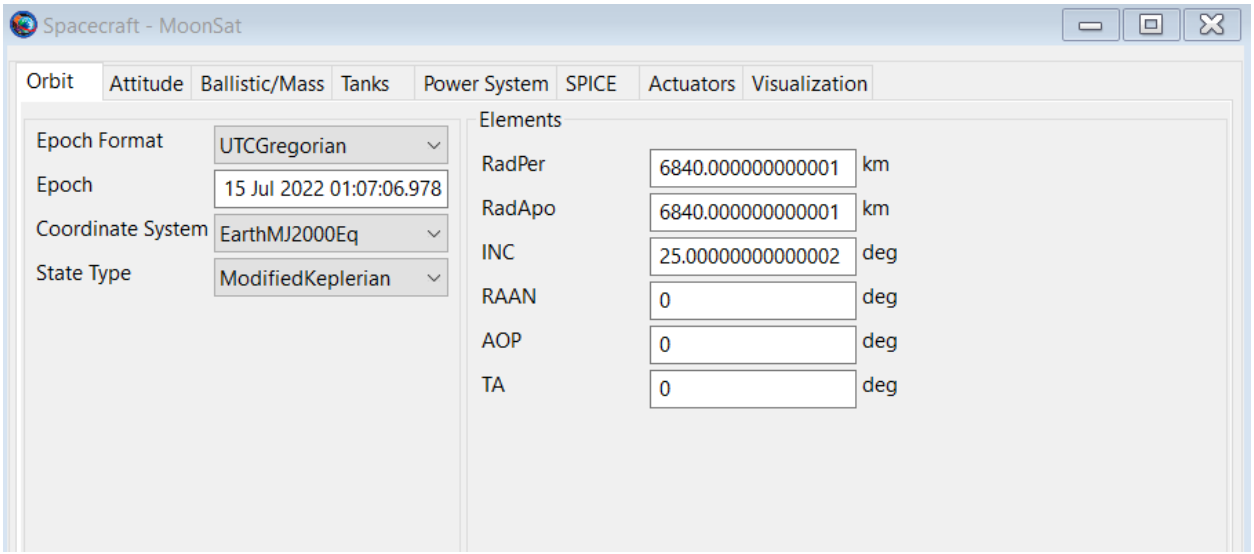


Figure 3.3 - Initial spacecraft property as defined in GMAT

The next step in GMAT is to define the two impulsive burns needed for the Earth-Moon transfer. They are named TOI (Transfer Orbit Insertion) and MOI (Moon Orbit Insertion) for this project. To rename the burns just right click on the burn and click rename or select the burn and click F2 key. The burns are impulsive burns, and the axes are set as VNB, which stands for Velocity-Normal-Binormal [15]. The origin is set at Earth for TOI burn, while for MOI burn, the origin is set at Luna (Moon). The initial value for the burns is zero because GMAT will find the optimal value for that transfer. The mass change section is untouched cause the goal of this project is to simulate the optimal transfer using GMAT.

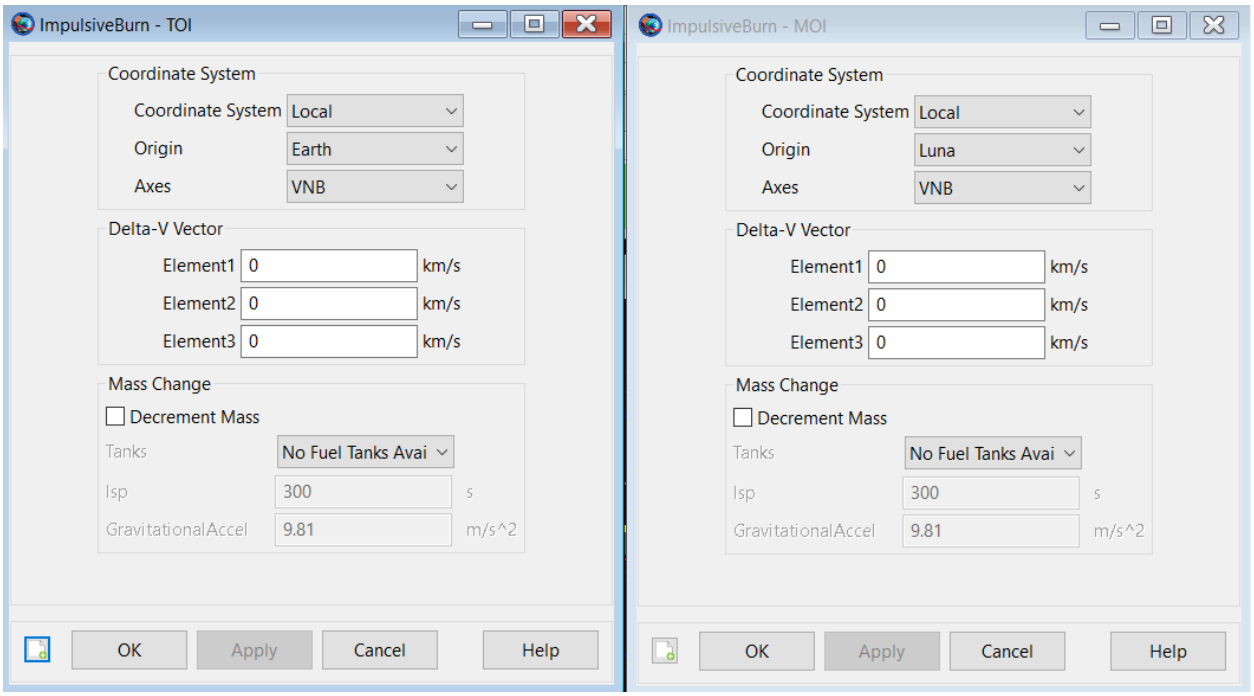


Figure 3.4 - TOI and MOI burn setup in GMAT

The next step is to model the propagator. A propagator is the GMAT component used to model spacecraft motion [16]. For this project, a numerical integrator-type propagator with a force model is used. Three propagators, NearEarth, EarthMoon, and NearMoon prop, are added to the GMAT project. As the name suggests, NearEarth propagator is used to model the spacecraft motion in LEO and only accounts for Earth gravitational force as a point mass. EarthMoon prop is used to find the ideal transfer trajectory between LEO to LMO, and spacecraft is subjected to both the Earth and Moon gravitational forces. And lastly, the NearMoonProp only accounts for Moon gravitational force on spacecraft and simulates orbit around LMO. Earth and Moon are modeled as point mass because by doing that, it allows users to only account for Earth and Moon gravitational attraction and not the perturbation caused by the gravity of Earth and Moon. The **primary body** are kept empties for all three propagators as a point mass model is implemented. (Figure 3.5) The integrator type to solve for the spacecraft's motion is selected as RungeKutta89 with an accuracy of 9.99e-12.

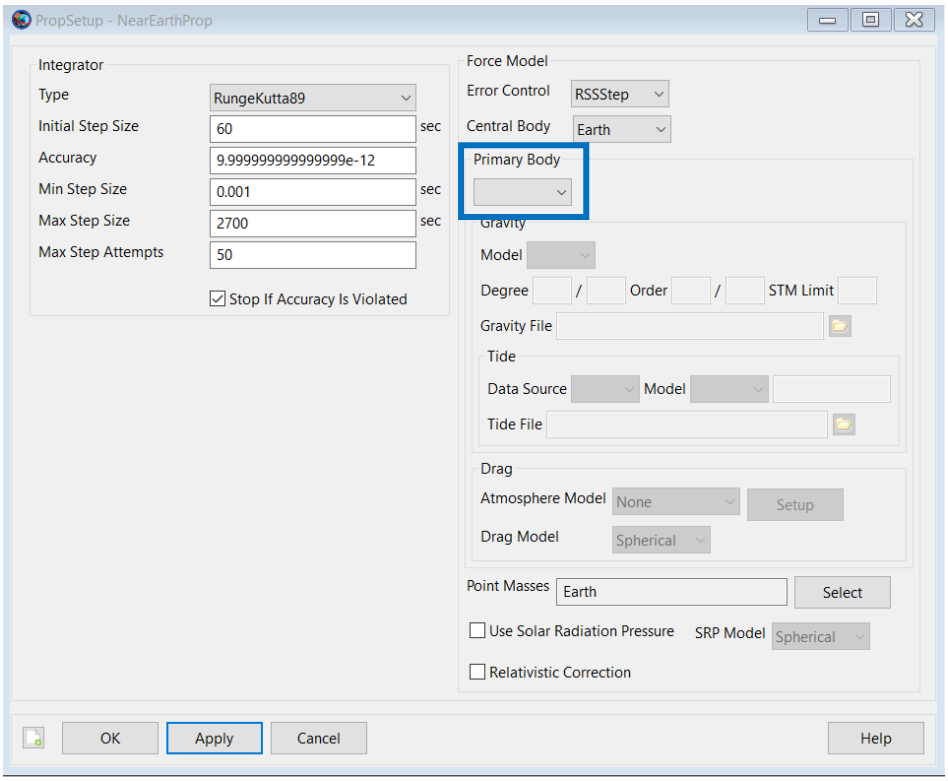


Figure 3.5 – NearEarthProp – GMAT properties only accounting for earth's gravity

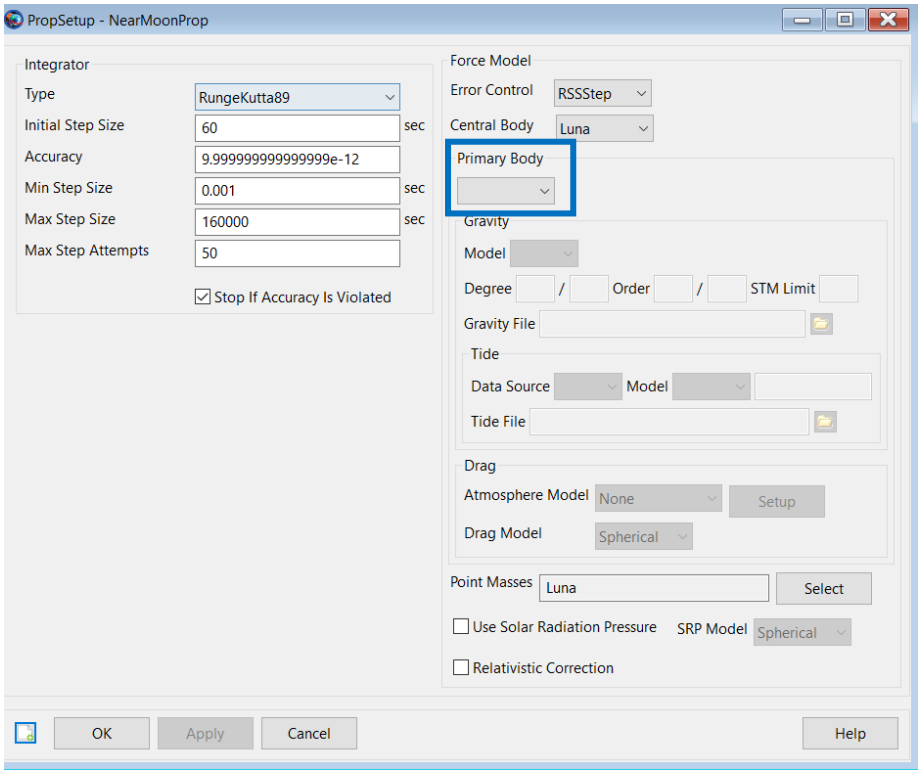


Figure 3.6 – NearMoonProp – GMAT properties only accounting for moon's gravity

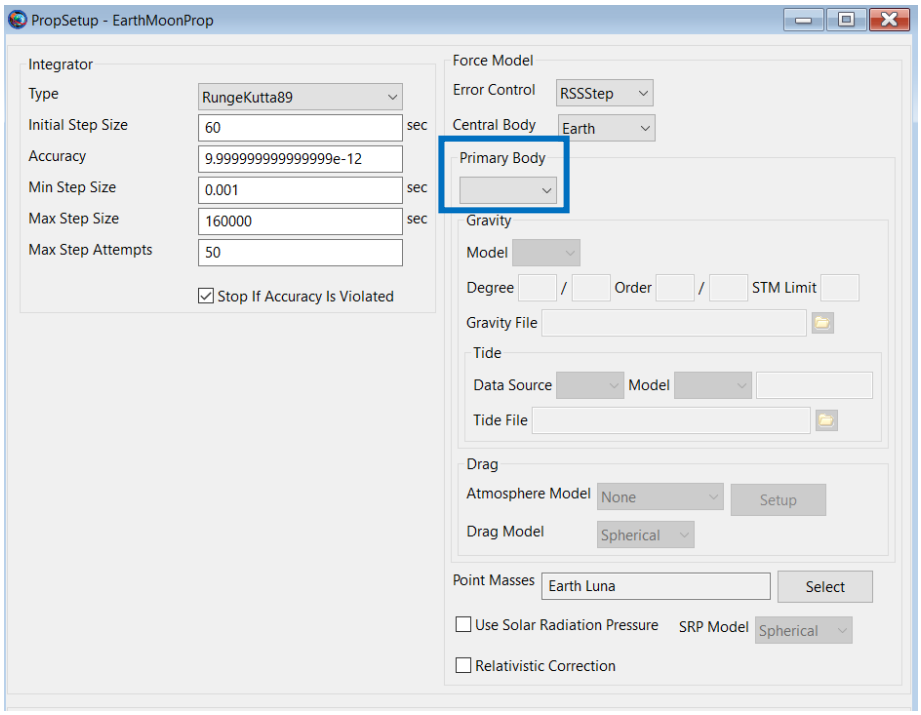


Figure 3.7 – EarthMoonProp – GMAT properties accounting for earth and moon gravity

To be able to visualize the transfer using GMAT, a couple of coordinate systems and orbit views are added. The first two coordinate systems added to GMAT are, EarthMoon rotation systems, and MoonEarth rotation system. An object referenced axis system is used to define this reference frames. An object referenced axis system is defined by the motion of one object with respect to another object. The Figure 3.8 defines six principal direction of an Object referenced axis system. One is the relative position of the secondary object with respect to the primary object, denoted by r which is expressed in the inertial frame. Second is the velocity of the secondary object w.r.t the primary expressed in the inertial frame by v. The third vector, n, normal to the direction of the motion which is calculated by n=r x v. [10]

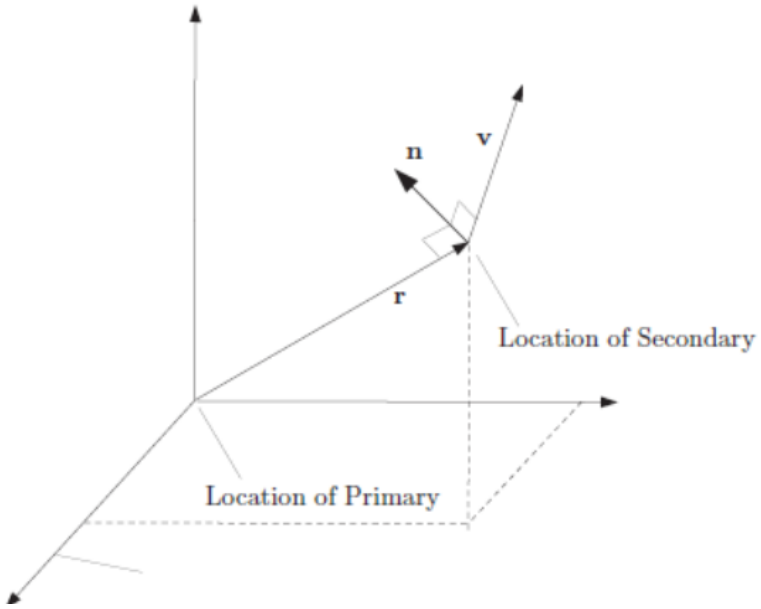


Figure 3.8 Visual representation of object referenced frame

Figure 3.9 below displays the axes type and primary and secondary bodies selected for the EathMoon and MoonEarth rotational frames.

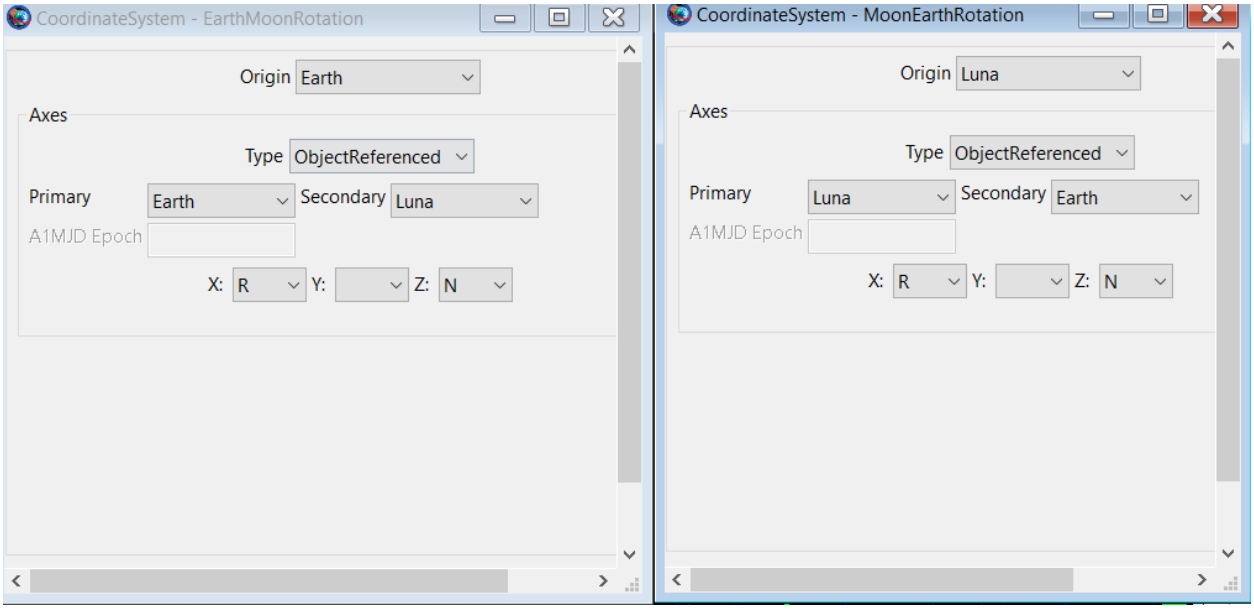


Figure 3.9 - EarthMoon and MoonEarth rotation frame as setup in GMAT

MoonMJ2000Eq and MoonInertial frame is also added to this project. Moon MJ2000 is used to solve the B-Plane problem, while the Moon inertial is used to visualize the orbit around the Moon.

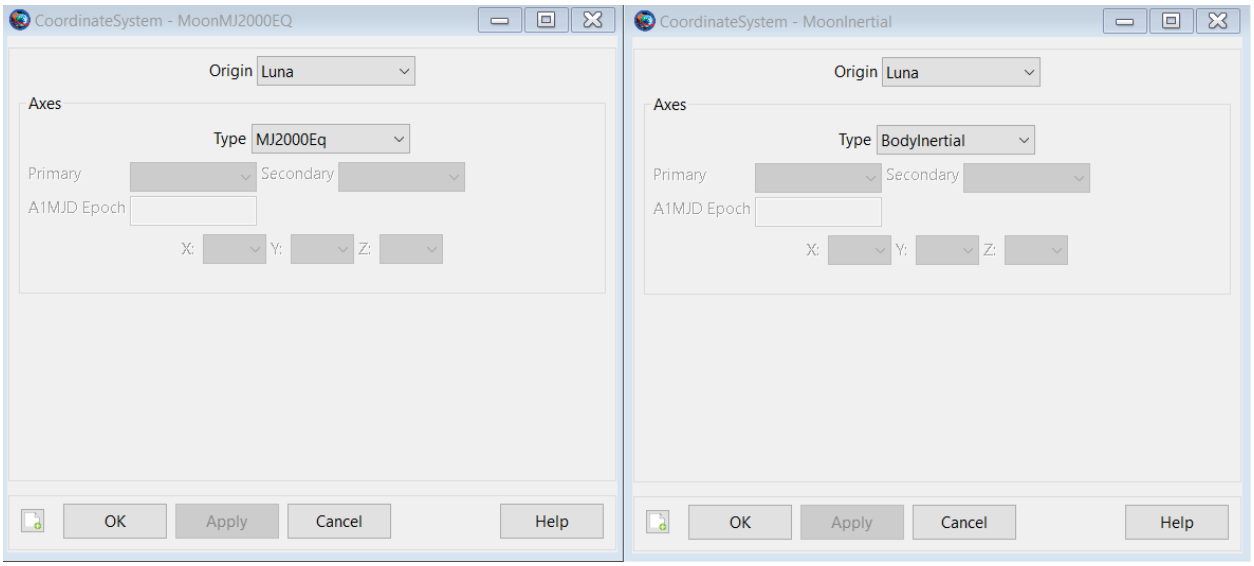


Figure 3.10 - MoonMJ2000EQ and MoonInertial reference frame as setup in GMAT

In addition to the Earth's inertial orbit view, two new orbit views are added for this project. One of which is 'EarthMoonRotationalView' rotation orbit view allows the user to see the whole transfer trajectory in Earth Moon rotational frame as.it uses 'EarthMoonRotational' coordinate frame. Figure 3.11 below shows the other orbit view parameter,

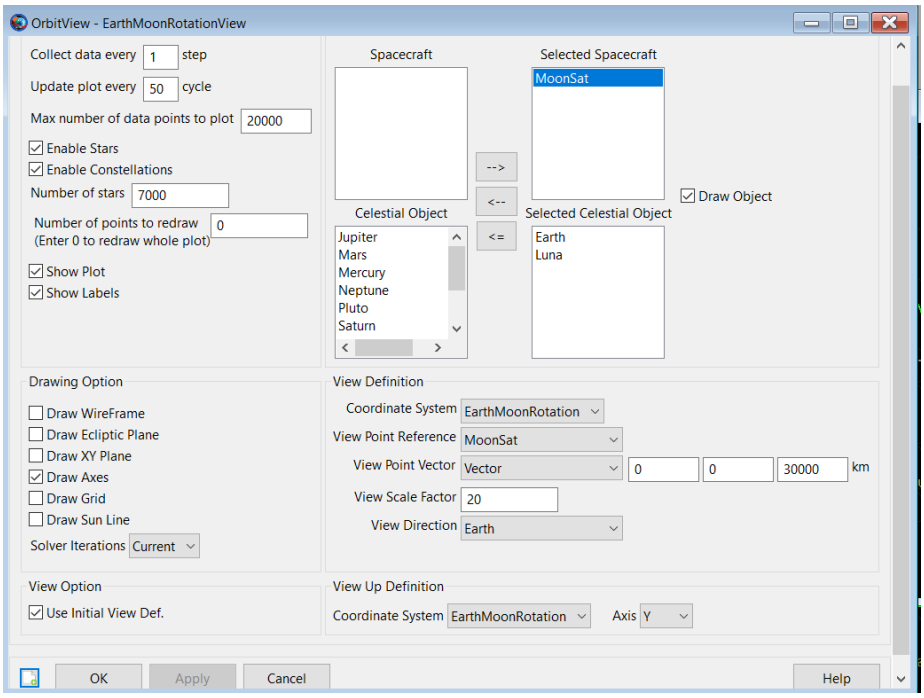


Figure 3.11 - EarthMoon rotational orbit view as setup in GMAT

'MooninertialOrbitalView' orbit view allows the user to see the spacecraft's orbit around the Moon. 'MoonInertial' coordinate system is used for this viewer and point of reference is defined as Moon. Figure 3.12 below displays other properties for the orbit viewer.

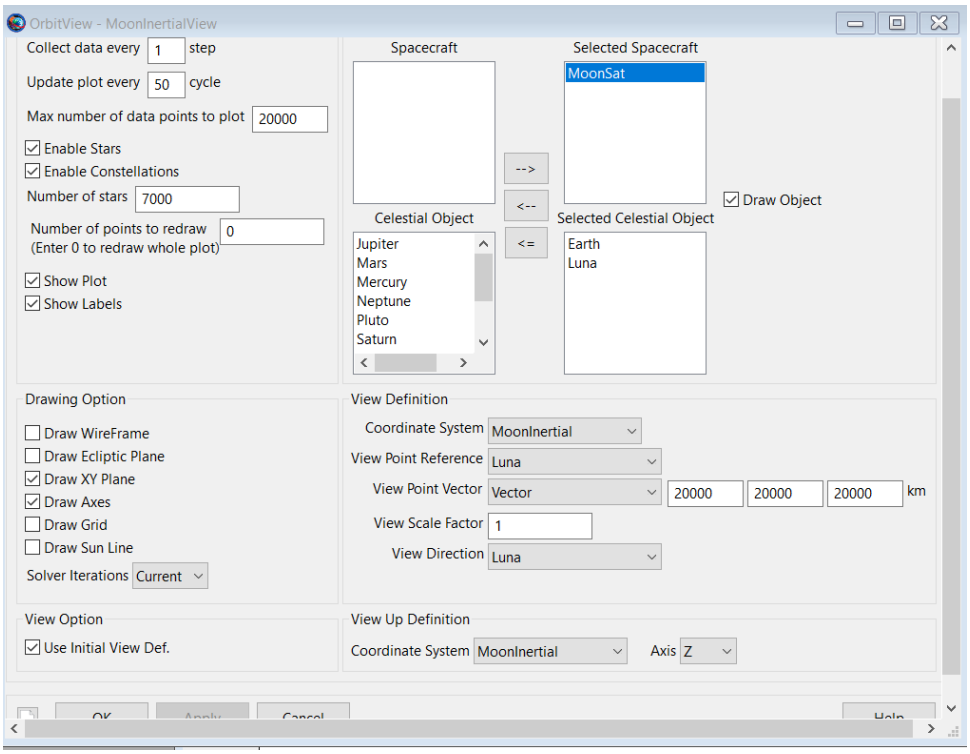


Figure 3.12 - Moon inertial orbit viewer as setup in GMAT

Spacecraft property, burns, propagators, outputs, and coordinate systems are all the GMAT components defined in the resources tab for this project. The mission tab contains the two target commands and the final orbit propagation around the Moon. 'Target' and 'EndTarget' commands in GMAT are used to solve condition(s) by varying one or more parameters. A Target sequence numerically solves a boundary value problem to determine the control variable's value required to satisfy the constraints. Control variables are defined using Vary commands, and constraints variables are defined using Achieve commands [16].

The first Target sequence is used to solve for the ΔV_{LEO} needed to get to the 100km altitude above Moon. The problem is set up such that B-Plan target, **B.R,** and **B.T** are set as constrain variables (achieve variable), and spacecraft RAAN, AOP, and TOI element one is set as the control variable (vary variable). Since GMAT is a hi-fidelity trajectory simulator, by varying the RAAN and AOP, the Earth to Moon transfer is converted into a 2-D problem similar to the MATLAB simulation. The key thing to note is the spacecraft motion at LEO is not simulated in GMAT, and because of that, RAAN and AOP variation does not cost any additional ΔV since GMAT varies the initial condition of the orbit. If the spacecraft orbit is simulated at LEO, changing the spacecraft trajectory to the ideal RAAN and AOP will add to the ΔV_{LEO} . TOI Element1, also known as the vector component (tangential) of the velocity, is allowed to be varied for this target sequence. **B.R** and **B.T** are set to 5090 km and 0 km, respectively. The values for which are found by trial and error and by examining the spacecraft altitude around the Moon. Then, the spacecraft is propagated to the desired radius around the Moon using the 'EarthMoonProp'. 'EarthMoonProp' accounts for both Earth and Moon effects on the spacecraft motion and finds the ideal trajectory based on the boundary value problem at hand. In summary, the following are the steps taken to set up the first Target sequence (Figure 3.21)

- **Create a Target Sequence**: Right click on 'Mission Sequence' → Append and → Target. Rename the target by right clicking or by pressing F2 to '*FindLunarTarget'*
- **Vary RAAN**: Setup a first control variable. Modify the lower and upper value and max step. To RAAN as an option click on Edit and select spacecraft.

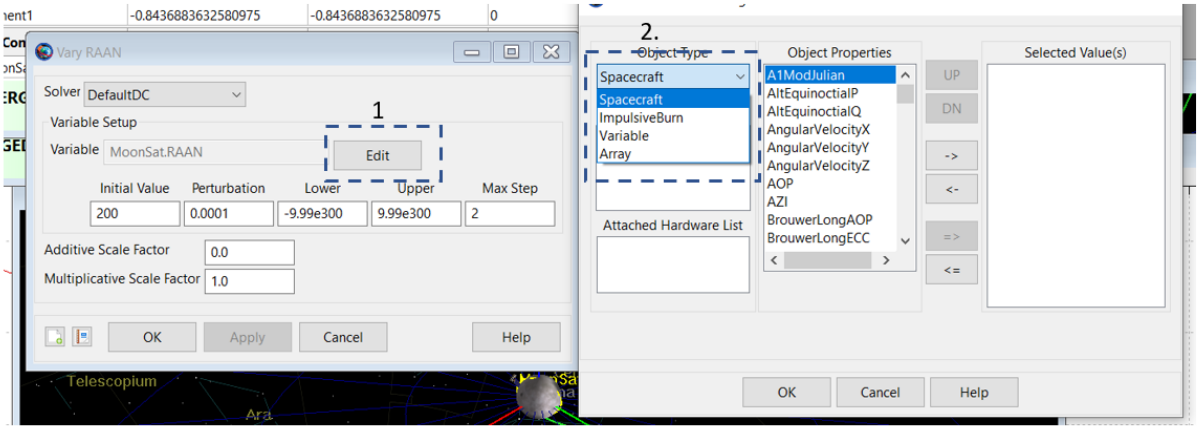


Figure 3.13 Initial step to setup RAAN as variable in GMAT

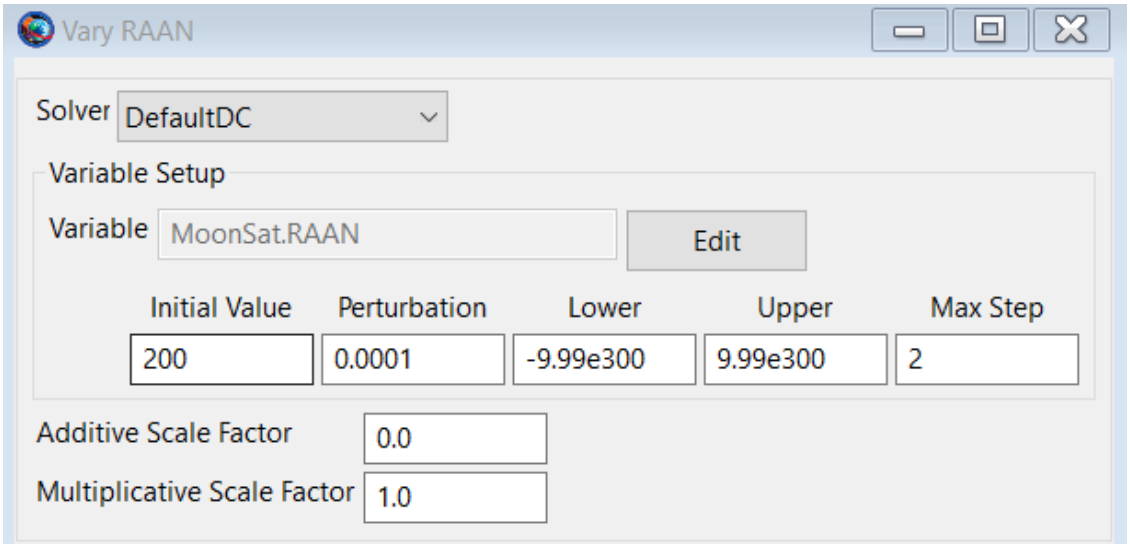


Figure 3.14 – Final parameter setup for varying RAAN

- **Vary AOP**: Similar to **Figure 3.13**, set up a control variable for AOP and update the upper and lower values and max step as shown below

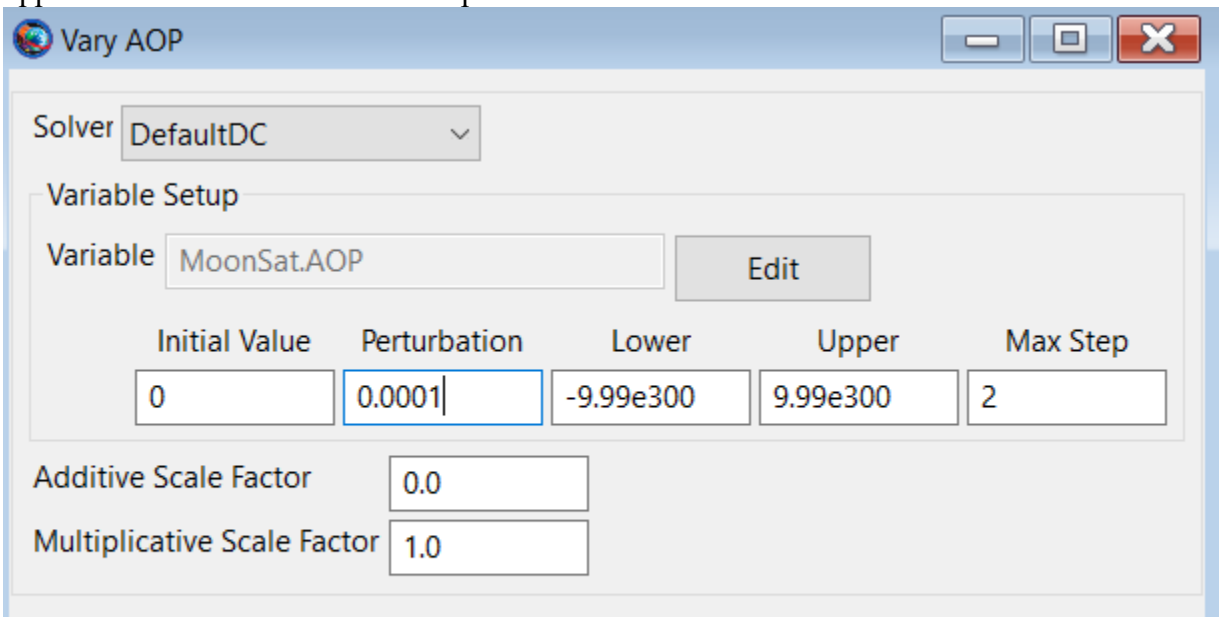


Figure 3.15 - Final parameter setup for varying AOP

- **Vary TOI Element1**: Setup a control variable for element 1 of TOI burn and update the upper and lower limit for the control variable.

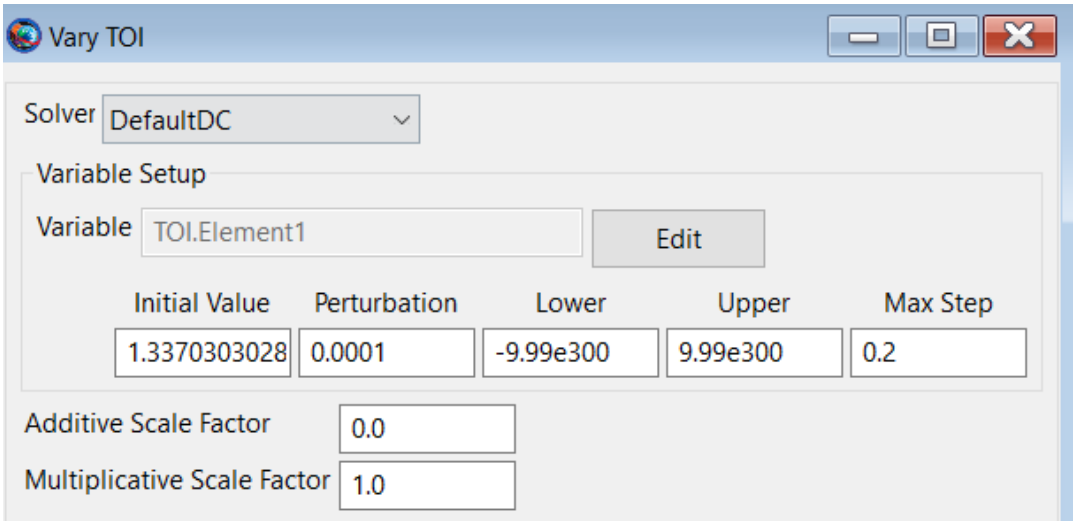


Figure 3.16 Final parameter setup for varying TOI element 1

- **Apply TOI**: Add a maneuver and select TOI burn and spacecraft name.

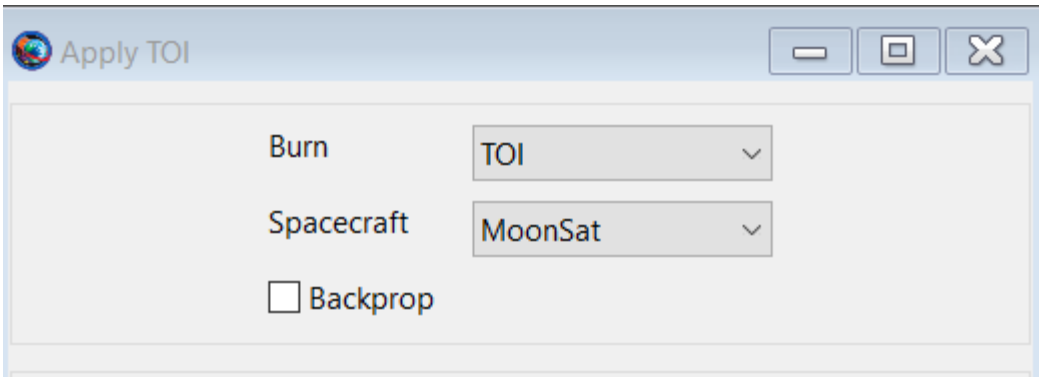


Figure 3.17 - Apply TOI maneuver to the spacecraft

- **Propagate to the Moon**: Add a propagator and propagate the spacecraft to the periapsis of the spacecraft in orbit around the Moon. 'EarthMoonProp' is selected as the propagator, and the stopping condition parameter is set at 'Luna.Periapsis'.

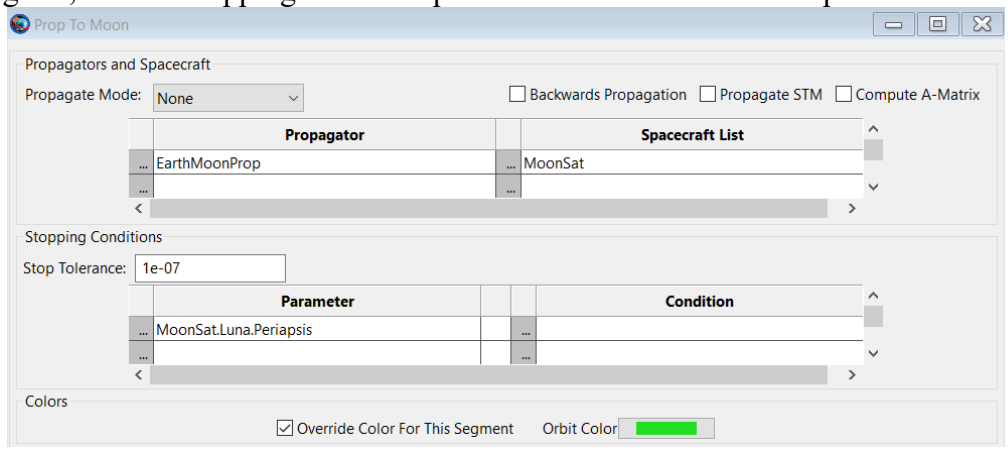


Figure 3.18 – GMAT propagate properties to propagate to moon periapsis

Figure 3.19, shows the steps needed to get Luna Periapsis.

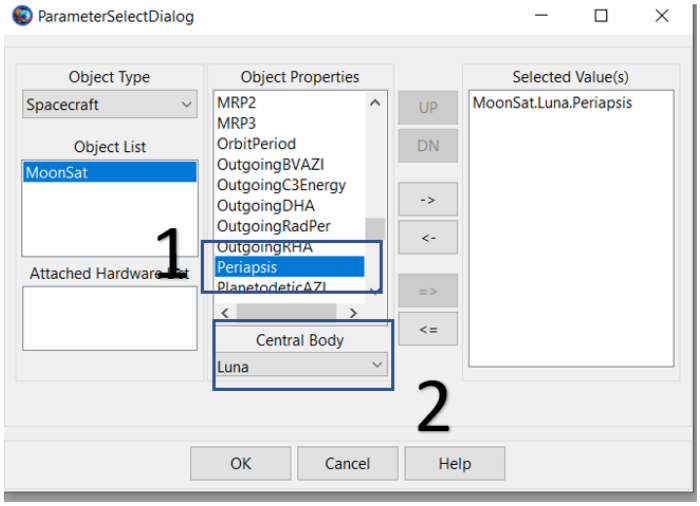


Figure 3.19 Initial Steps to get luna periapsis as GMAT propagate parameter

- **$\overrightarrow{B.R}$ and $\overrightarrow{B.T}$ Constrain variable**: Two constrain variables are added to achieve desired transfer. The **B.R** and **B.T** are defined using the MoonMJ2000EQ coordinate system, and the default tolerance is kept for these two constrained variables.

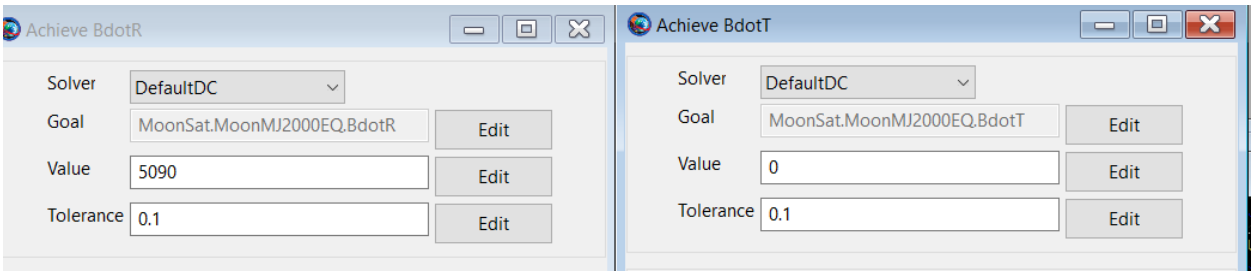


Figure 3.20 - B.R and **B.T** constrain variables

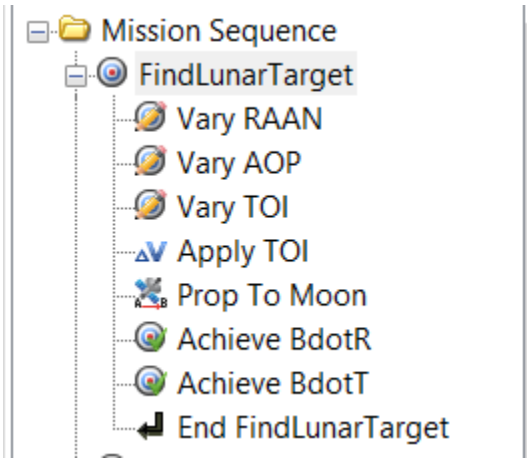


Figure 3.21 – Overview of find lunar target sequence

The second target sequence is set up to find the ΔV_{LMO} needed to circularize the orbit around Moon. This burn takes place at the periapsis radius since the last propagator stops once the spacecraft reaches the periapsis radius around the Moon. Unlike the previous target, this one only has one control variable, the MOI burn's tangential velocity (element 1). There are two constraints, achieve the radius of apoapsis of 1837 km (similar apoapsis radius as the MATLAB) and achieve 0 eccentricities around Moon. Moon has an equatorial radius of 1738.1 km and polar radius of 1736.0 km [14], and since GMAT accounts for that radial fluctuations, the tolerance for 'AchieveRapo' is kept at 2 km while eccentricity is maintained at 0.001. Following are the steps are taken to set up the second target, 'CircularOrbitAroundMoon'.

- **Create a Target Sequence**: Right click on 'Mission Sequence' → Append and → Target. Rename the target by right clicking or by pressing F2 to '*CircularOrbitAroundMoon'*
- **Vary MOI**: Vary element 1 of the MOI burn. A key thing to note is to vary MOI initial value is kept at -1, and this is because GMAT is throwing an unknown error when the initial value is kept at zero. Upper and lower limits are also updated. For this control variable, upper value can be kept at zero since it's known that the burn has to be retrograde burn, but it's kept at a positive infinity for consistency.

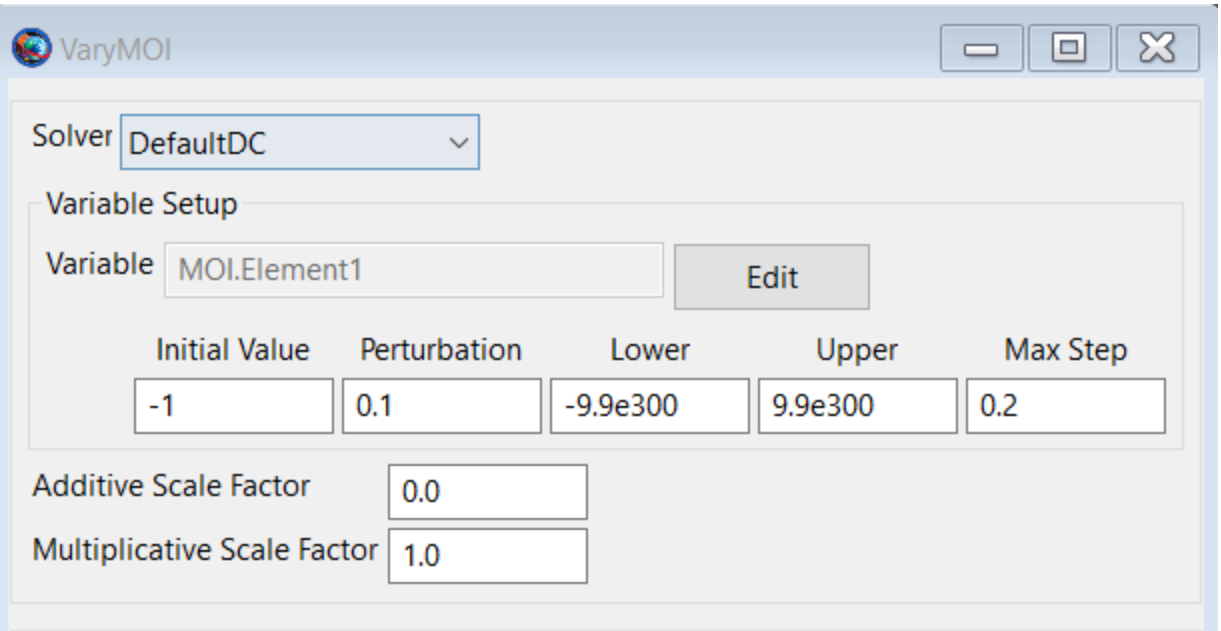


Figure 3.22 – Final variable setup to vary MOI element 1

- **Apply MOI:** The next step is adding the maneuver by selecting the MOI burn and the correct spacecraft.

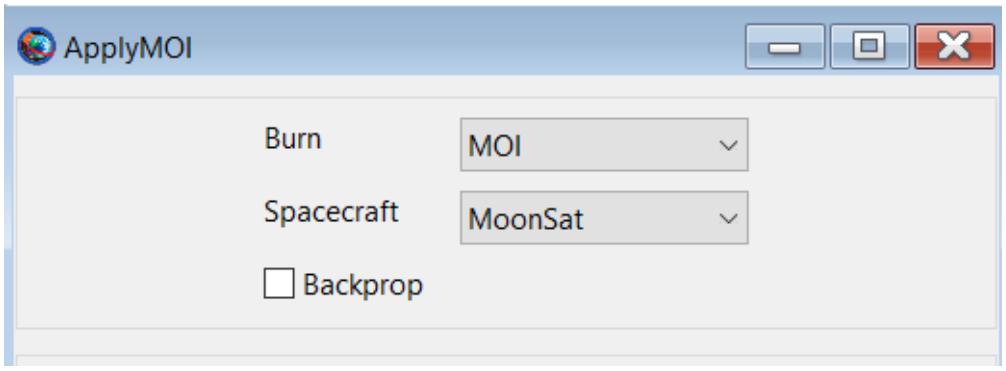


Figure 3.23 - Apply the MOI maneuver

- **PropToApoapsis**: A propagator is added to propagate the spacecraft to the apoapsis of the Moon. 'NearMoonProp' is used to propagate the spacecraft, and the stopping parameter is set to Luna. Apoapsis.

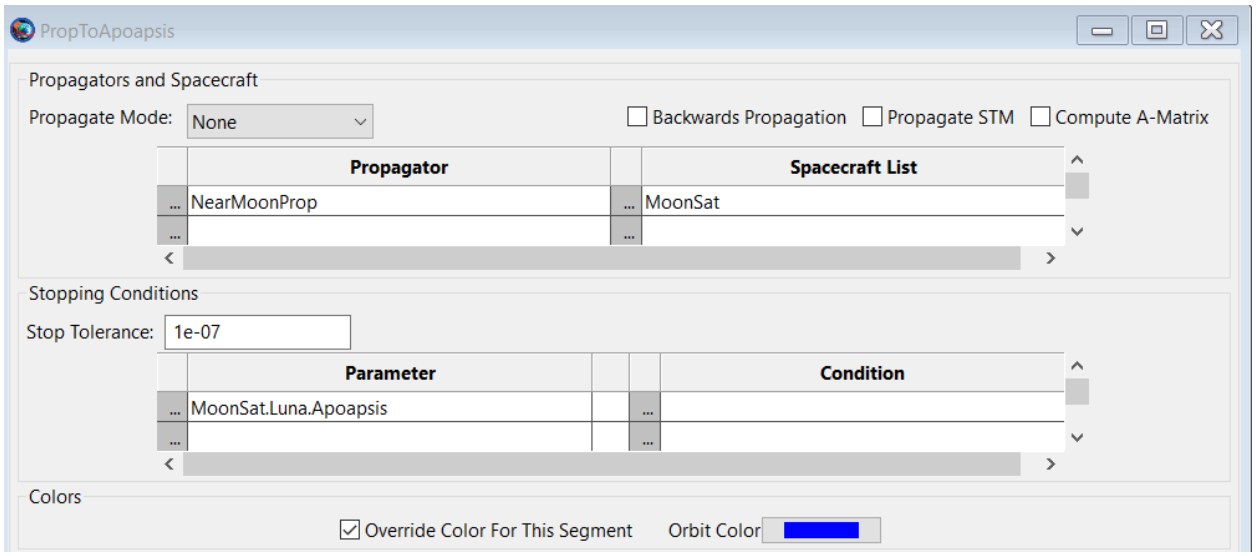


Figure 3.24 -GMAT propagator setup to propagate to moon apoapsis

- **Achieve Radius of Apoapsis**: The first constrain is set to achieve the radius of Apoapsis of 1837 km, i.e., approximately 100 km orbit around the Moon. The key thing to note is the tolerance is set to 2 km because of the difference in radius between Moon's equator and Moon's poles.
- **Achieve Eccentricity**: The second constrain is set to achieve the eccentricity of 0, i.e., a circular orbit around the Moon.

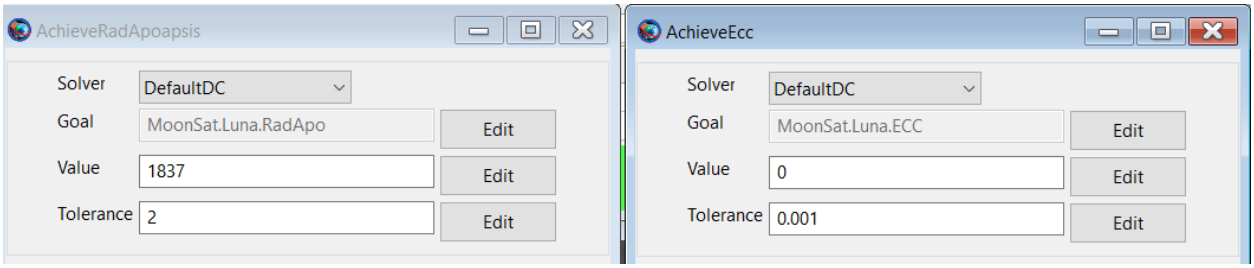


Figure 3.25 – GMAT setup to achieve desired moon orbit

Lastly, the spacecraft orbit around the Moon is simulated for another day using a NearMoon propagator to visualize the circular orbit at LMO. Figure 3.27 summarizes the CircularOrbitAroundMoon target sequence and final LMO orbit propagator.

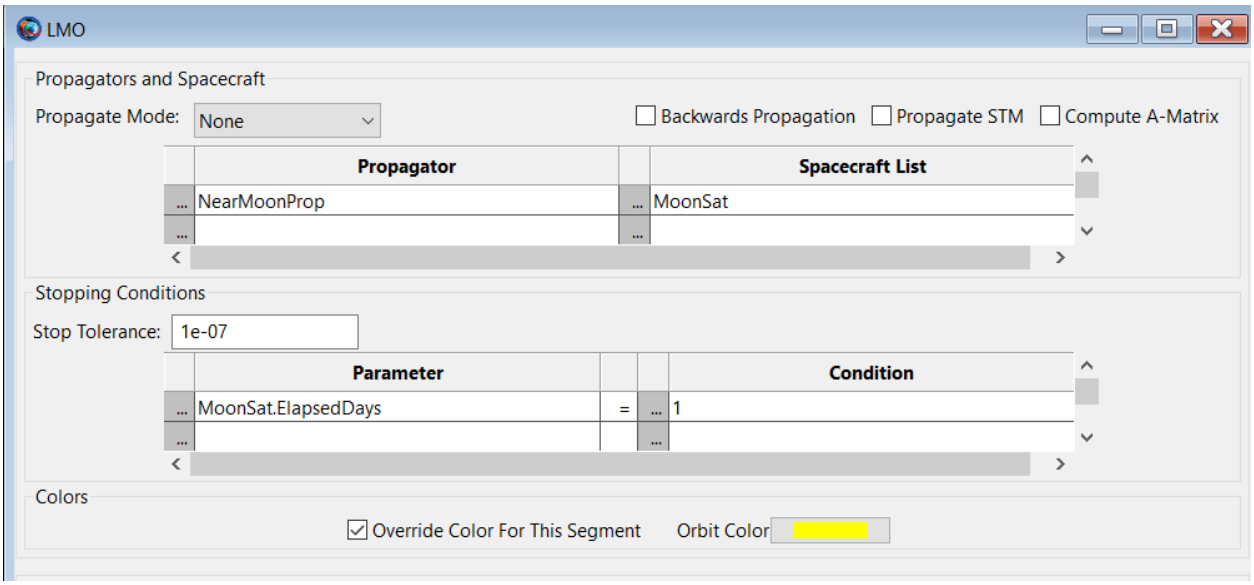


Figure 3.26 – Another day of moon orbit view setup

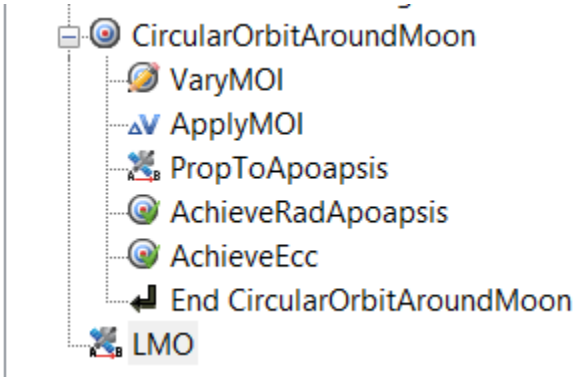


Figure 3.27 - Mission sequence for circularization orbit around moon

4. Results

This section covers the result for MATLAB simulation, GMAT simulation and compares them to the previously published data.

4.1 MATLAB Result

Equations 28.1 and 28.2 are integrated using the native MATLAB ODE45 function to plot the spacecraft's trajectory. Constants used for these scripts are listed in Table 3.1 The $\Delta V_{LEO} = 3.069 \frac{km}{s}$ and phase angle $\delta = -117.52°$ used for this problem are taken from reference 9. The ODE function is integrated until the following orbit condition is met.

Spacecraft Final position,

$$x_{final} = x_{moon} + r_{Moon} + SpacecraftAltitude \quad (4.1)$$

To find x_{Moon} equation 12 is used and r_{Moon} is a constant defined in Table 3.1, and spacecraft altitude is defined as 100 km above the surface of the Moon. Figure 4.1 shows the overall orbit trajectory as seen from the Z-axis of the Earth-Moon rotational frame.

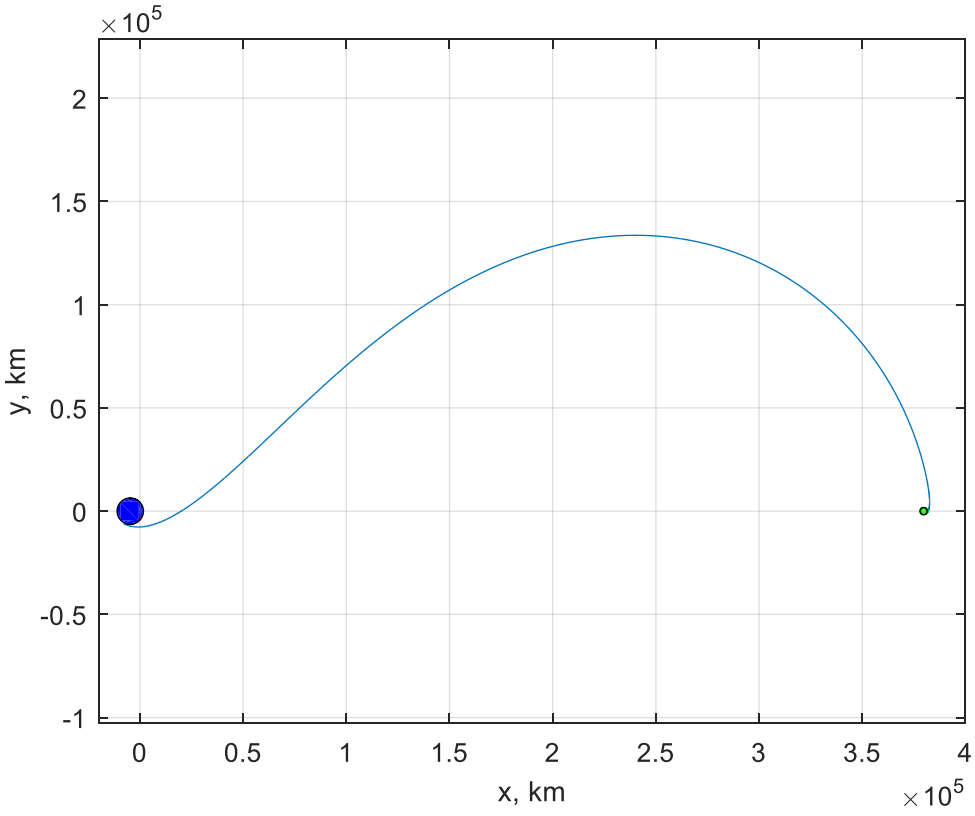


Figure 4.1 - Spacecraft trajectory calculated using MATLAB

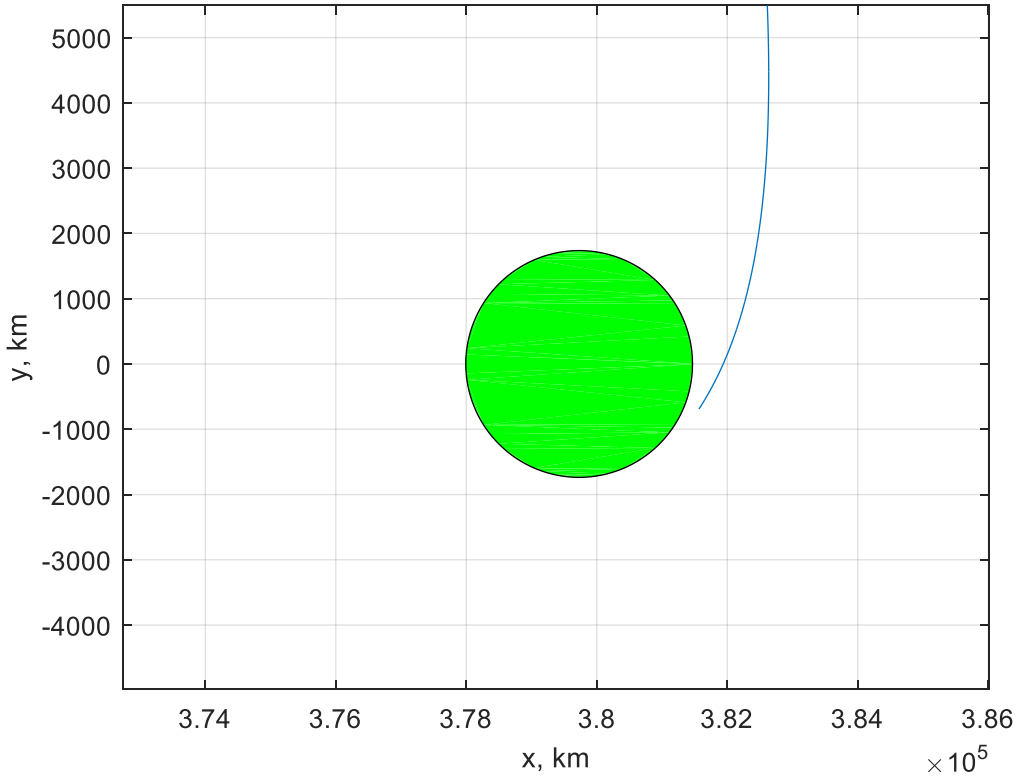


Figure 4.2 - Spacecraft approach at Moon using MATLAB

ΔV_{LMO} is calculated using equation 4.2. The total ΔV_{Total} cost for the transfer between LEO to LMO is calculated,

$$\Delta V_{Total} = \Delta V_{LEO} + \Delta V_{LMO} \tag{4.2}$$

$$\Delta V_{Total} = 3.069 + 0.7718 = 3.8408 \frac{km}{s}$$

The transfer for LEO to LMO calculated using MATLAB takes 4.41 days.

4.2 GMAT Results

GMAT is used to simulate a similar trajectory from LEO to LMO. Instead of inputting initial ΔV_{LEO} and a phase angle δ, the trajectory is solved using the GMAT native RungKutta89 solver by setting up a boundary value problem using the GMAT Target sequences. Overall orbit trajectory seen from the Z-axis of the Earth-Moon rotational frame is shown in Figure 4.3. The trajectory seems to follow a similar shape as displayed in Figure 4.1.

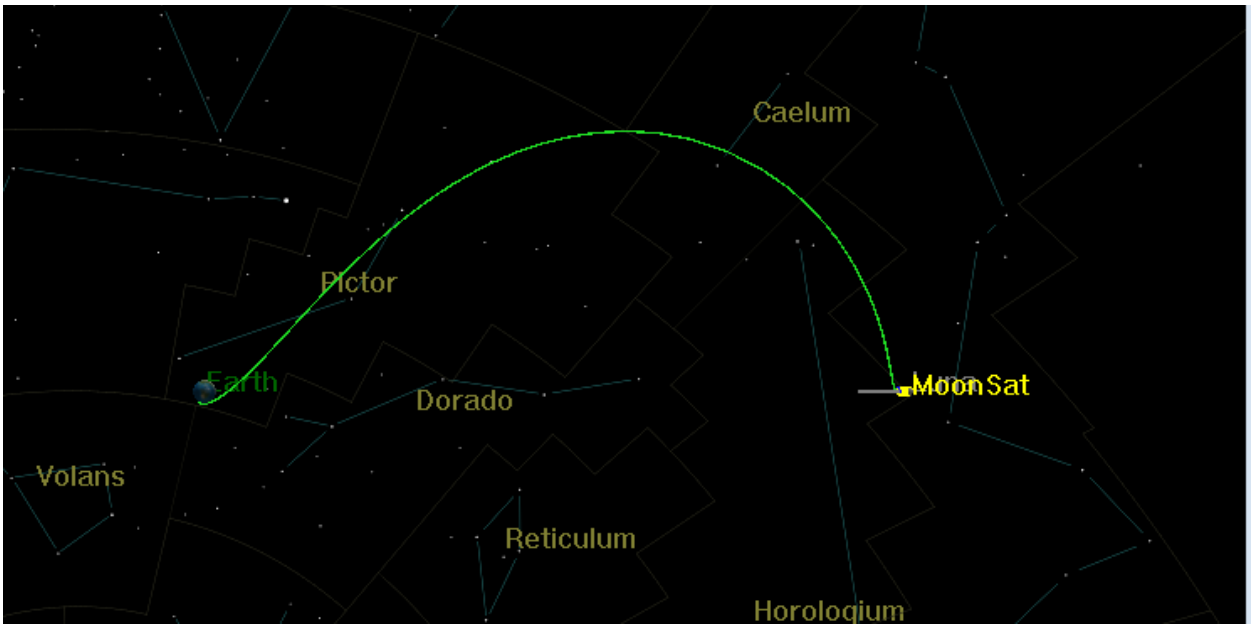


Figure 4.3 - GMAT Satellite overall trajectory

ΔV_{LEO} calculated by GMAT to transfer to spacecraft from LEO to ~100 km altitude above Moon is 3.0681, very close to the ΔV_{LEO} value used in the MATLAB simulation.

Solver Window - Target 'FineLunarTarget' DefaultDC {SolveMode = Solve, ExitMode = DiscardA...

Control Variable	Current Value	Last Value	Difference
MoonSat.RAAN	194.1588159424351	194.1588159424351	-2.842170943040401e-14
MoonSat.AOP	-8.96758828634216	-8.96758828634216	0
TOI.Element1	3.068165035580054	3.068165035580054	0

Constraints	Desired	Achieved	Difference
(==) MoonSat.MoonMJ2000E(	5090	5089.997739910057	-0.002260089942865307
(==) MoonSat.MoonMJ2000E(	0	0.002988065325325806	0.002988065325325806

CONVERGED

Figure 4.4 - Solver window for TOI (Transfer Orbit Insertion)

ΔV_{LMO} calculated using GMAT to circularize the orbit at LMO is 0.8436 $\frac{km}{s}$. The ΔV_{Total} calculated using GMAT is,

$$\Delta V_{TotalGMAT} = \Delta V_{\text{LEOGMAT}} + \Delta V_{LMOGMAT} \quad 4.3$$
$$\Delta V_{TotalGMAT} = 3.0681 + 0.8436 = 3.9117$$

Solver Window - Target 'CircularOrbitAroundMoon' DefaultDC {SolveMode = Solve, ExitMode ...

Control Variable	Current Value	Last Value	Difference
MOI.Element1	-0.8436834554086824	-0.8436834554086824	0
Constraints	**Desired**	**Achieved**	**Difference**
(==) MoonSat.Luna.RadApo	1837	1837.496218440698	0.4962184406981578
(==) MoonSat.Luna.ECC	0	4.726587719259852e-05	4.726587719259852e-05

CONVERGED

Figure 4.5 - Solver window for MOI (Moon Orbit Insertion)

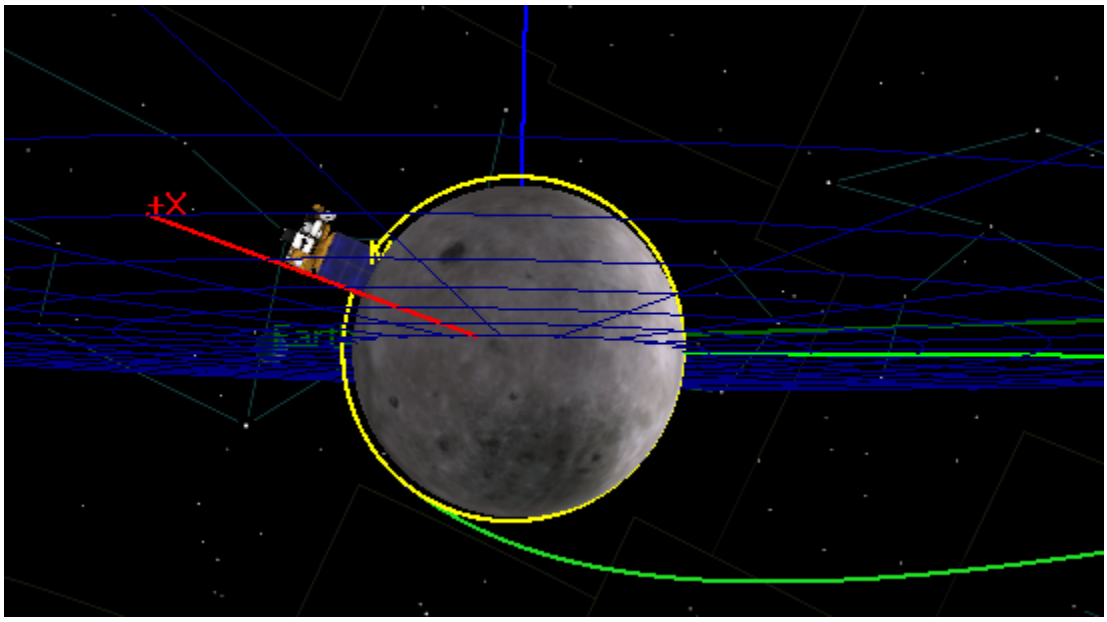


Figure 4.6 - Orbit insertion and orbit around the moon

The transfer time needed to send the spacecraft from circular LEO to circular LMO is 4.1 days.

4.3 Result Summary and Comparison

Table 4.1 below summarizes the LEO to LMO transfer result between MATLAB simulation, GMAT simulation, and Leonardo and Pontani published data [9].

Table 4.1 - LEO-LMO result summary

Variable	MATLAB	GMAT	Published Data	Note
$\Delta V_{LEO}\ \frac{km}{s}$	3.069	3.0681	3.069	For the Matlab Simulation ΔV_{LEO} is used as an initial condition
$\Delta V_{LMO}\ \frac{km}{s}$	0.7718	0.8436	0.8160	

TOF (days)	4.41	4.1	4.5	Published data do not explicitly state their TOF is 4.5 days but hints that the result is similar to the published data in reference 8

Percent error between each result is calculated in Table 4.2. Since GMAT is the hi-fidelity simulation, data calculated using GMAT is considered as the most accurate result, and percent error is calculated as follows,

$$\%Error = \frac{|GMAT\ Result - MATLAB\ Result|}{GMAT\ Result} \times 100 \quad 4.4$$

And

$$\%Error = \frac{|GMAT\ Result - Published\ Result|}{GMAT\ Result} \times 100 \quad 4.5$$

Table 4.2 - Simulation percentage error

Variable	MATLAB %Error	Published Result %Error
$\Delta V_{LEO}\ \frac{km}{s}$	0.029 %	0.029 %
$\Delta V_{LMO}\ \frac{km}{s}$	9.30 %	3.57 %
TOF (days)	7.56 %	9.75 %

5. Lesson Learned

The following section covers the lesson learned while working on the project.

5.1 MATLAB Simulation /Canonical Form

The orbit trajectory in MATLB is numerically solved in the engineering units and not in canonical form. Since the orbital parameter are in high order numbers i.e. 10^5 or more, minor fluctuations cause a drastically different answer. For example, the orbit simulation in MATLAB is done by using μ_{Earth}=398600 $\frac{km^3}{\sec^2}$ which provides a pretty accurate trajectory when compared with the previously published data and GMAT. But if the same simulation is run with the $\mu_{Earth} = G * m_E = 3.9836 \times 10^5 \frac{km^3}{\sec^2}$ the result is much different. Figure 5.1 below displays the spacecraft motion under both initial conditions. Using the Canonical unit system mitigates these problems since constants are no longer huge numbers.

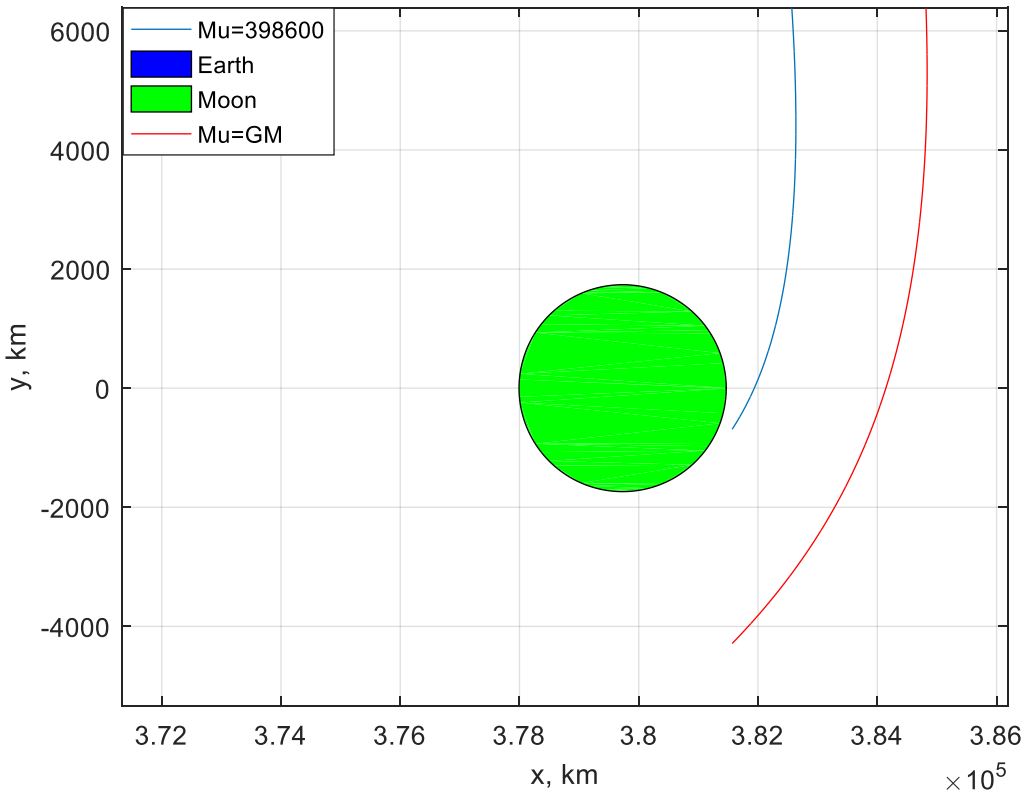


Figure 5.1 - Difference in result because of minor difference in μ_{Earth} variable

5.2 Calculating initial **B.R** and **B.T** value

For this project, **B.R** and **B.T** values used in GMAT simulation for the B-Plane transfer is assessed using trial and error. The values were constantly edited until the desired altitude around the moon is achieved. This section covers how to calculate the initial **B.R** and **B.T** value mathematically by knowing desired position and velocity vector. To calculate the B-Plane values the orbit position and velocity needs to be in 3-dimensions and in body centric inertial frame, MCI (Moon Centric Inertial) for this transfer. Figure 5.2 below defines the MCI coordinate systems. Z_{MCI} passing through Lunar north pole, X_{MCI} passing through the vernal equinox and Y_{MCI} passing through the Lunar equator such that it satisfy the right hand rule.

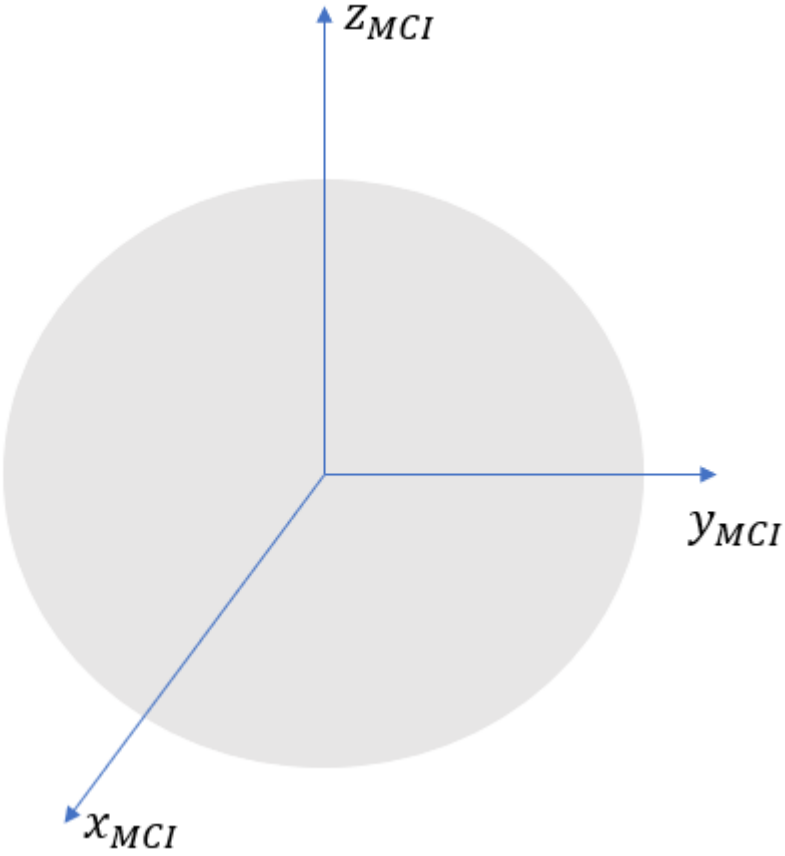


Figure 5.2 Moon centered inertial frame

5.2.1: Sample Calculation

Following position and velocity vector are used,

$$[rx, ry, rz] = [0, 1.837, -689] \text{ km} \quad (5.1)$$

$$[vx, vy, vz] = [0, 1.2970, 2.007] \text{ km/s} \quad (5.2)$$

Both position and velocity vector initial guess are taken from the CR3BP MATLAB code. Since the CR3BP is solved in 2-D, following assumption is made,

X position in CR3BP= Y position in MCI frame

Y position in CR3BP = Z position in MCI frame

And X position in MCI frame = 0 = Spacecraft is in Polar orbit with inclination =90° and RAAN = 90°

$$\vec{h} = \vec{r} \times \vec{v} = [4.5689, 0\ , 0]\frac{km^2}{m} \tag{5.3}$$

Semiparameter

$$p = \frac{h^2}{\mu} = 4257.8\ km \tag{5.4}$$

Semimajor axis

$$a = \frac{r}{2 - \frac{rv^2}{\mu}} = -7135.0\ km \tag{5.5}$$

Eccentricity,

$$e = \sqrt{1 - \frac{p}{a}} = 1.2636 \tag{5.6}$$

True anomaly (cosine and sine)

$$cos\theta = \frac{p - r}{e * r} = 0.9260\ rad \tag{5.7}$$

$$sin\theta = \frac{\dot{r}h}{e * \mu} = 0.3774\ rad \tag{5.8}$$

B-Plane magnitude

$$b = \sqrt{p * |a|} = 5511.7\ km \tag{5.9}$$

Fundamental unit vectors

$$\hat{z} = \frac{r\vec{v} - \dot{r}\vec{r}}{h} = [0\ , 0.3512\ , 0.9363] \tag{5.10}$$

$$\hat{p} = cos\theta\ \hat{r} - sin\theta\ \hat{z} = [0\ , 0.7345\ , -0.6786] \tag{5.11}$$

$$\hat{q} = sin\theta\ \hat{r} + cos\theta\ \hat{z} = [0, 0.6786, 0.7345] \tag{5.12}$$

where,

$$\hat{r} = \frac{\vec{r}}{|r|} and\ \dot{r} = \frac{\vec{r}.\vec{v}}{|r|} \tag{5.13}$$

S Vector

$$\vec{S} = -\frac{a}{\sqrt{a^2 + b^2}}\hat{p} + \frac{b}{\sqrt{a^2 + b^2}}\hat{q} = [0\ , 0.9961, -0.0880] \tag{5.14}$$

B- Vector

$$\vec{B} = \frac{b^2}{\sqrt{a^2 + b^2}}\ \hat{p} + \frac{ab}{\sqrt{a^2 + b^2}}\ \hat{q} = [0\ , -485.0 - 5490.3]\ km \tag{5.15}$$

T – vector

$$T = \frac{\left(S_y^2, -S_x^2, 0\right)^T}{\sqrt{S_x^2 + S_y^2}} = [1, 0, 0] \quad 5.16$$

R -Vector

$$\vec{R} = \vec{S} \times \vec{T} = [0, -0.0880, -0.9961] \quad 5.17$$

$$\vec{B}.\vec{T} = 0\ km \quad 5.18$$

$$\vec{B}.\vec{R} = 5511.7\ km \quad 5.19$$

The $\vec{B}.\vec{R}$ calculated mathematically is 9% higher than the one estimated using the trial-and-error method on GMAT. This is because the initial value of position and velocity used for the calculation came by solving the equation motion in CR3BP. Mathematically calculated $\vec{B}.\vec{R}$ and $\vec{B}.\vec{T}$ still can be used as the initial guess on the GMAT.

6. Conclusion

Bi-Impulsive LEO-LMO transfer is examined in this paper. Trajectory is generated using both MATLAB and NASA GMAT. GMAT simulation result is than compared with the MATLAB result and previously published data for the similar trajectory. Percent error between the GMAT and the MATLAB simulation for the ΔV_{LMO} is less than 10% and the ΔV_{LEO} is less than 0.03%. Similarly, percent error between the GMAT and published data is 0.03% and 3.57% for ΔV_{LEO} and ΔV_{LMO}respectively. Trajectory in MATLAB is simulated using engineering units which could be a possible source of the error as minor fluctuation in constant values, such as gravitational constant of the Earth, can cause a large integral error.

Reference

[1] Nasa. “NASA’s lunar exploration program overview.” 2020.

[2] Topputo, F. “On optimal two-impulse earth-moon transfers in a four-body model.”

[3] Bate, R. R., Mueller, D. D., and White, J. E. *FUNDAMENTALS OF ASTRODYNAMICS*. Dover Publication INC., 1971.

[4] Riche, L. J., Colton, G. M., and Guillory, T. A. *Flight Plan Apollo 11* . Houston Texas, 1969.

[5] Belbruno, E. A., and Miller, J. K. “Sun-Perturbed earth-to-moon transfers with ballistic capture.” *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, 1993, pp. 770–775. https://doi.org/10.2514/3.21079.

[6] Qi, Y., and Xu, S. “Minimum Δv for the transfer to permanent lunar orbits with hyperbolic approach.” *Acta Astronautica*, Vol. 119, No. September 2017, 2016, pp. 183–195. https://doi.org/10.1016/j.actaastro.2015.11.016.

[7] Mengali, G., and Quarta, A. A. “Optimization of biimpulsive trajectories in the earth-moon Restricted three-body system.” *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 2, 2005, pp. 209–216. https://doi.org/10.2514/1.7702.

[8] Miele, A., and Mancuso, S. “Optimal trajectories for earth–moon–earth flight.” *Acta Astronautica*, Vol. 49, No. 2, 2001, pp. 59–71. https://doi.org/10.1016/S0094-5765(01)00007-8.

[9] Leonardi, E. M., and Pontani, M. “Optimal two- and three-dimensional earth–moon orbit transfers.” *Aerotecnica Missili & Spazio*, Vol. 99, No. 3, 2020, pp. 195–202. https://doi.org/10.1007/s42496-020-00046-2.

[10] *GMAT User Guide R2020a*. 2019.

[11] Cho, D. H., Chung, Y., and Bang, H. "Trajectory correction maneuver design using an improved b-plane targeting method." *Acta Astronautica*, Vol. 72, 2012, pp. 47–61. https://doi.org/10.1016/j.actaastro.2011.11.009.

[12] Kizner, W. "A method of describing miss distance for lunar and interplanetary trajectories." Vol. 1, 1959, pp. 105–112.

[13] *GMAT Mathematical Specification*. Greenbelt MD, 2020.

[14] Williams, D. Moon Fact Sheet. *NASA Goddard Space Flight Center* . https://nssdc.gsfc.nasa.gov/planetary/factsheet/moonfact.html. Accessed Dec. 11, 2021.

[15] ImpulsiveBurn. http://gmat.sourceforge.net/doc/R2016a/html/ImpulsiveBurn.html. Accessed Dec. 11, 2021.

[16] Guide, U. *GMAT User Guide R2016a*.

Appendix A. Three Body problem using MATLAB

AE 295B - Earth to Moon transfer using a 3 body Equation of Motion

Jay Mehta Advisor: Jeanine Hunter AE 295B

```
clear all
close all
clc
```

Global Variables

```
global x_Earth x_Moon mu_Earth mu_Moon mu_System omega Ratio_Earth Ratio_Moon DistanceEarthMoon
RadiusMoon
```

Constants

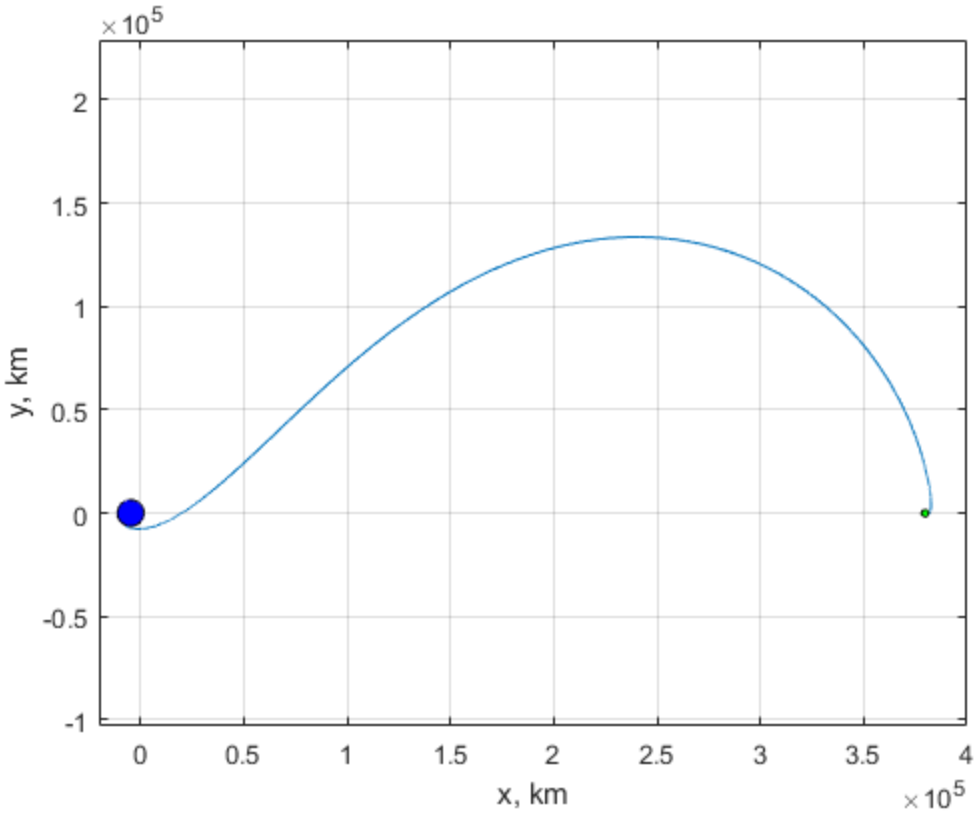
```
GConstant=6.67e-20; % Gravitational constant km^3/kg*sec^2
RadiusEarth=6378;   %Radius of Earth in Km
RadiusMoon=1737;    %Radius of Moon in Km
DistanceEarthMoon=384400; %Distance of Moon from Earth in km
MassEarth=5.9724e24;        %Mass of Earth in Kg
MassMoon=0.07346e24;         %Mass of Moon in Kg
TotalMass=MassEarth+MassMoon; %Total mass of the system since spacecraft mass is negligible in Kg
Ratio_Earth=MassEarth/TotalMass; %Dimensionless Mass of the Earth
Ratio_Moon=MassMoon/TotalMass; %Dimensionless Mass of Moon
mu_Earth=GConstant*MassEarth;  %Gravitational Constant of Earth
%mu_Moon=GConstant*MassMoon;    %Gravitational Constant of Moon
mu_Moon=4903.02;
mu_Earth=398600;  %Gravitational Constant of Earth

mu_System=mu_Earth+mu_Moon;    %Gravitational Constant of whole system
omega=sqrt(mu_System/DistanceEarthMoon^3); %Angular velocity of Earth and moon around its
barycenter
x_Earth=-Ratio_Moon*DistanceEarthMoon; % distance of Earth from the barycenter in km
x_Moon=Ratio_Earth*DistanceEarthMoon;  %distance of Moon from the barycenter in Km

%Initial Condition
SpacecraftAltitude=463; %Altitude of the spacecraft around the Earth in km
delta_ang=-117.52; %Phase Angle; Value chosen from previously published Data
deltaV=3.069; % Delta V applied at LEO, value chosen from previously published Data

SpaceCraftRadius=RadiusEarth+SpacecraftAltitude; %LEO Orbit Radius
v0=sqrt(mu_Earth/SpaceCraftRadius)+deltaV; %Inital Velocity at LEO
t0=0; %Intial Time
tf=5*24*60*60; %Max Time for ODE 45 solver
x0=SpaceCraftRadius*cosd(delta_ang)+x_Earth;
y0=SpaceCraftRadius*sind(delta_ang);
```

```
vx0 = (omega*SpaceCraftRadius-v0)*sind(delta_ang);
vy0 = (v0-omega*SpaceCraftRadius)*cosd(delta_ang);

IC=[x0;y0;vx0;vy0]; %Inital Condition passed to ODE45 solver

options = odeset('AbsTol', 1e-29, 'RelTol', 1e-29,'Events',@events) %Eventfunction sets a
stopping condition for the ODE solver
[t,f] =ode45(@EOM, [t0 tf], IC,options);
xSpaceCraft = f(:,1);
ySpaceCraft = f(:,2);
vxSpaceCraft = f(:,3);
vySpaceCraft =f(:,4);
xFinal = xSpaceCraft(end);
yFinal = ySpaceCraft(end);
vxFinal = vxSpaceCraft(end);
vyFinal = vySpaceCraft(end);
df = norm([xFinal - x_Moon, yFinal - 0]) - RadiusMoon;
vf = norm([vxFinal, vyFinal]);
```

```
options =

  struct with fields:

              AbsTol: 1.0000e-29
                 BDF: []
              Events: @events
         InitialStep: []
            Jacobian: []
           JConstant: []
            JPattern: []
                Mass: []
        MassSingular: []
            MaxOrder: []
             MaxStep: []
         NonNegative: []
         NormControl: []
           OutputFcn: []
           OutputSel: []
              Refine: []
              RelTol: 1.0000e-29
               Stats: []
          Vectorized: []
    MStateDependence: []
           MvPattern: []
        InitialSlope: []

Warning: RelTol has been increased to 2.22045e-14.
```

CoastingArch Plots

```
figure,
plot(xSpaceCraft, ySpaceCraft)
%Set plot display parameters
xmin = -20.e3; xmax = 4.e5;
ymin = -20.e3; ymax = 1.e5;
axis([xmin xmax ymin ymax])
axis equal
xlabel('x, km'); ylabel('y, km')
grid on
hold on
earth = circle(x_Earth, 0, RadiusEarth);
moon = circle(x_Moon, 0, RadiusMoon);
orbit=circle(x_Moon,0,RadiusMoon+100);
fill(earth(:,1), earth(:,2),'b');
fill( moon(:,1), moon(:,2),'g');
%plot(orbit(:,1),orbit(:,2),'r');
```



Orbit of the Moon

```
theta=asind(yFinal/1837)
theta2=acosd((xFinal-x_Moon)/1837);

xdot=sqrt(mu_Moon/1837)*sind(theta)+omega*yFinal
ydot=-sqrt(mu_Moon/1837)*cosd(theta2)-omega*(xFinal-x_Moon)
```

```
deltaVLMP=sqrt((xdot-vxFinal)^2+(ydot-vyFinal)^2)
V=norm([vxFinal,vyFinal])
V-deltaVLMP

j=deltaV+deltaVLMP
```

```
theta =

  -22.0582


xdot =

   -0.6154


ydot =

   -1.6386


deltaVLMP =

    0.7718


V =

    2.3843


ans =

    1.6125


j =

    3.8408
```

Published with MATLAB® R2016b

Appendix B. GMAT three body script

%General Mission Analysis Tool(GMAT) Script

%Created: 2021-11-12 18:14:37

```
%----------------------------------------
%---------- Spacecraft
%----------------------------------------

Create Spacecraft MoonSat;
GMAT MoonSat.DateFormat = UTCGregorian;
GMAT MoonSat.Epoch = '15 Jul 2022 01:07:06.978';
GMAT MoonSat.CoordinateSystem = EarthMJ2000Eq;
GMAT MoonSat.DisplayStateType = ModifiedKeplerian;
GMAT MoonSat.RadPer = 6840.99999999999;
GMAT MoonSat.RadApo = 6840.999999999995;
GMAT MoonSat.INC = 25.00000000000002;
GMAT MoonSat.RAAN = 199.9999999999999;
GMAT MoonSat.AOP = 0;
GMAT MoonSat.TA = 8.537699999999958;
GMAT MoonSat.DryMass = 850;
GMAT MoonSat.Cd = 2.2;
GMAT MoonSat.Cr = 1.8;
GMAT MoonSat.DragArea = 15;
GMAT MoonSat.SRPArea = 1;
GMAT MoonSat.SPADDragScaleFactor = 1;
GMAT MoonSat.SPADSRPScaleFactor = 1;
GMAT MoonSat.NAIFId = -10000001;
GMAT MoonSat.NAIFIdReferenceFrame = -9000001;
GMAT MoonSat.OrbitColor = Red;
GMAT MoonSat.TargetColor = Teal;
```

```
GMAT MoonSat.OrbitErrorCovariance = [ 1e+70 0 0 0 0 0 ; 0 1e+70 0 0 0 0 ; 0 0 1e+70 0 0 0 ;
0 0 0 1e+70 0 0 ; 0 0 0 0 1e+70 0 ; 0 0 0 0 0 1e+70 ];

GMAT MoonSat.CdSigma = 1e+70;

GMAT MoonSat.CrSigma = 1e+70;

GMAT MoonSat.Id = 'SatId';

GMAT MoonSat.Attitude = CoordinateSystemFixed;

GMAT MoonSat.SPADSRPInterpolationMethod = Bilinear;

GMAT MoonSat.SPADSRPScaleFactorSigma = 1e+70;

GMAT MoonSat.SPADDragInterpolationMethod = Bilinear;

GMAT MoonSat.SPADDragScaleFactorSigma = 1e+70;

GMAT MoonSat.ModelFile = 'aura.3ds';

GMAT MoonSat.ModelOffsetX = 0;

GMAT MoonSat.ModelOffsetY = 0;

GMAT MoonSat.ModelOffsetZ = 0;

GMAT MoonSat.ModelRotationX = 0;

GMAT MoonSat.ModelRotationY = 0;

GMAT MoonSat.ModelRotationZ = 0;

GMAT MoonSat.ModelScale = 1;

GMAT MoonSat.AttitudeDisplayStateType = 'Quaternion';

GMAT MoonSat.AttitudeRateDisplayStateType = 'AngularVelocity';

GMAT MoonSat.AttitudeCoordinateSystem = EarthMJ2000Eq;

GMAT MoonSat.EulerAngleSequence = '321';
```

```
%---------------------------------------
%---------- ForceModels
%---------------------------------------

Create ForceModel EarthMoonProp_ForceModel;
GMAT EarthMoonProp_ForceModel.CentralBody = Earth;
GMAT EarthMoonProp_ForceModel.PointMasses = {Earth, Luna};
GMAT EarthMoonProp_ForceModel.Drag = None;
GMAT EarthMoonProp_ForceModel.SRP = Off;
GMAT EarthMoonProp_ForceModel.RelativisticCorrection = Off;
GMAT EarthMoonProp_ForceModel.ErrorControl = RSSStep;

Create ForceModel NearMoonProp_ForceModel;
GMAT NearMoonProp_ForceModel.CentralBody = Luna;
GMAT NearMoonProp_ForceModel.PointMasses = {Luna};
GMAT NearMoonProp_ForceModel.Drag = None;
GMAT NearMoonProp_ForceModel.SRP = Off;
GMAT NearMoonProp_ForceModel.RelativisticCorrection = Off;
GMAT NearMoonProp_ForceModel.ErrorControl = RSSStep;

Create ForceModel NearEarthProp_ForceModel;
GMAT NearEarthProp_ForceModel.CentralBody = Earth;
GMAT NearEarthProp_ForceModel.PointMasses = {Earth};
```

```
GMAT NearEarthProp_ForceModel.Drag = None;
GMAT NearEarthProp_ForceModel.SRP = On;
GMAT NearEarthProp_ForceModel.RelativisticCorrection = Off;
GMAT NearEarthProp_ForceModel.ErrorControl = RSSStep;
GMAT NearEarthProp_ForceModel.SRP.Flux = 1367;
GMAT NearEarthProp_ForceModel.SRP.SRPModel = Spherical;
GMAT NearEarthProp_ForceModel.SRP.Nominal_Sun = 149597870.691;

%----------------------------------------
%---------- Propagators
%----------------------------------------

Create Propagator EarthMoonProp;
GMAT EarthMoonProp.FM = EarthMoonProp_ForceModel;
GMAT EarthMoonProp.Type = RungeKutta89;
GMAT EarthMoonProp.InitialStepSize = 60;
GMAT EarthMoonProp.Accuracy = 9.999999999999999e-12;
GMAT EarthMoonProp.MinStep = 0.001;
GMAT EarthMoonProp.MaxStep = 160000;
GMAT EarthMoonProp.MaxStepAttempts = 50;
GMAT EarthMoonProp.StopIfAccuracyIsViolated = true;

Create Propagator NearMoonProp;
GMAT NearMoonProp.FM = NearMoonProp_ForceModel;
GMAT NearMoonProp.Type = RungeKutta89;
GMAT NearMoonProp.InitialStepSize = 60;
GMAT NearMoonProp.Accuracy = 9.999999999999999e-12;
GMAT NearMoonProp.MinStep = 0.001;
```

```
GMAT NearMoonProp.MaxStep = 160000;
GMAT NearMoonProp.MaxStepAttempts = 50;
GMAT NearMoonProp.StopIfAccuracyIsViolated = true;

Create Propagator NearEarthProp;
GMAT NearEarthProp.FM = NearEarthProp_ForceModel;
GMAT NearEarthProp.Type = RungeKutta89;
GMAT NearEarthProp.InitialStepSize = 60;
GMAT NearEarthProp.Accuracy = 9.999999999999999e-12;
GMAT NearEarthProp.MinStep = 0.001;
GMAT NearEarthProp.MaxStep = 2700;
GMAT NearEarthProp.MaxStepAttempts = 50;
GMAT NearEarthProp.StopIfAccuracyIsViolated = true;

%----------------------------------------
%---------- Burns
%----------------------------------------

Create ImpulsiveBurn TOI;
GMAT TOI.CoordinateSystem = Local;
GMAT TOI.Origin = Earth;
GMAT TOI.Axes = VNB;
GMAT TOI.Element1 = 0;
GMAT TOI.Element2 = 0;
GMAT TOI.Element3 = 0;
GMAT TOI.DecrementMass = false;
GMAT TOI.Isp = 300;
GMAT TOI.GravitationalAccel = 9.81;
```

```
Create ImpulsiveBurn MOI;
GMAT MOI.CoordinateSystem = Local;
GMAT MOI.Origin = Luna;
GMAT MOI.Axes = VNB;
GMAT MOI.Element1 = 0;
GMAT MOI.Element2 = 0;
GMAT MOI.Element3 = 0;
GMAT MOI.DecrementMass = false;
GMAT MOI.Isp = 300;
GMAT MOI.GravitationalAccel = 9.81;

%----------------------------------------
%---------- Coordinate Systems
%----------------------------------------

Create CoordinateSystem EarthMoonRotation;
GMAT EarthMoonRotation.Origin = Earth;
GMAT EarthMoonRotation.Axes = ObjectReferenced;
GMAT EarthMoonRotation.XAxis = R;
GMAT EarthMoonRotation.ZAxis = N;
GMAT EarthMoonRotation.Primary = Earth;
GMAT EarthMoonRotation.Secondary = Luna;

Create CoordinateSystem MoonEarthRotation;
GMAT MoonEarthRotation.Origin = Luna;
GMAT MoonEarthRotation.Axes = ObjectReferenced;
GMAT MoonEarthRotation.XAxis = R;
```

```
GMAT MoonEarthRotation.ZAxis = N;
GMAT MoonEarthRotation.Primary = Luna;
GMAT MoonEarthRotation.Secondary = Earth;

Create CoordinateSystem MoonInertial;
GMAT MoonInertial.Origin = Luna;
GMAT MoonInertial.Axes = BodyInertial;

Create CoordinateSystem MoonMJ2000EQ;
GMAT MoonMJ2000EQ.Origin = Luna;
GMAT MoonMJ2000EQ.Axes = MJ2000Eq;

Create CoordinateSystem LunaFixed;
GMAT LunaFixed.Origin = Luna;
GMAT LunaFixed.Axes = BodyFixed;

%----------------------------------------
%---------- Solvers
%----------------------------------------

Create DifferentialCorrector DefaultDC;
GMAT DefaultDC.ShowProgress = true;
GMAT DefaultDC.ReportStyle = Normal;
GMAT DefaultDC.ReportFile = 'DifferentialCorrectorDefaultDC.data';
GMAT DefaultDC.MaximumIterations = 150;
GMAT DefaultDC.DerivativeMethod = ForwardDifference;
GMAT DefaultDC.Algorithm = NewtonRaphson;
```

```
%----------------------------------------
%---------- Subscribers
%----------------------------------------

Create OrbitView EarthMoonRotationView;
GMAT EarthMoonRotationView.SolverIterations = Current;
GMAT EarthMoonRotationView.UpperLeft = [ 0.1708253358925144 0.1523341523341523 ];
GMAT EarthMoonRotationView.Size = [ 0.6301983365323096 0.6904176904176904 ];
GMAT EarthMoonRotationView.RelativeZOrder = 5;
GMAT EarthMoonRotationView.Maximized = false;
GMAT EarthMoonRotationView.Add = {MoonSat, Earth, Luna};
GMAT EarthMoonRotationView.CoordinateSystem = EarthMoonRotation;
GMAT EarthMoonRotationView.DrawObject = [ true true true ];
GMAT EarthMoonRotationView.DataCollectFrequency = 1;
GMAT EarthMoonRotationView.UpdatePlotFrequency = 50;
GMAT EarthMoonRotationView.NumPointsToRedraw = 0;
GMAT EarthMoonRotationView.ShowPlot = true;
GMAT EarthMoonRotationView.MaxPlotPoints = 20000;
GMAT EarthMoonRotationView.ShowLabels = true;
GMAT EarthMoonRotationView.ViewPointReference = Earth;
GMAT EarthMoonRotationView.ViewPointVector = [ 0 0 30000 ];
GMAT EarthMoonRotationView.ViewDirection = Earth;
GMAT EarthMoonRotationView.ViewScaleFactor = 20;
GMAT EarthMoonRotationView.ViewUpCoordinateSystem = EarthMoonRotation;
GMAT EarthMoonRotationView.ViewUpAxis = Y;
GMAT EarthMoonRotationView.EclipticPlane = Off;
GMAT EarthMoonRotationView.XYPlane = Off;
GMAT EarthMoonRotationView.WireFrame = Off;
```

```
GMAT EarthMoonRotationView.Axes = Off;
GMAT EarthMoonRotationView.Grid = Off;
GMAT EarthMoonRotationView.SunLine = Off;
GMAT EarthMoonRotationView.UseInitialView = On;
GMAT EarthMoonRotationView.StarCount = 7000;
GMAT EarthMoonRotationView.EnableStars = On;
GMAT EarthMoonRotationView.EnableConstellations = On;

Create OrbitView MoonInertialView;
GMAT MoonInertialView.SolverIterations = Current;
GMAT MoonInertialView.UpperLeft = [ 0.0671785028790787 0.343980343980344 ];
GMAT MoonInertialView.Size = [ 0.7485604606525912 0.5651105651105651 ];
GMAT MoonInertialView.RelativeZOrder = 21;
GMAT MoonInertialView.Maximized = false;
GMAT MoonInertialView.Add = {MoonSat, Earth, Luna};
GMAT MoonInertialView.CoordinateSystem = MoonInertial;
GMAT MoonInertialView.DrawObject = [ true true true ];
GMAT MoonInertialView.DataCollectFrequency = 1;
GMAT MoonInertialView.UpdatePlotFrequency = 50;
GMAT MoonInertialView.NumPointsToRedraw = 0;
GMAT MoonInertialView.ShowPlot = true;
GMAT MoonInertialView.MaxPlotPoints = 20000;
GMAT MoonInertialView.ShowLabels = true;
GMAT MoonInertialView.ViewPointReference = Luna;
GMAT MoonInertialView.ViewPointVector = [ 20000 20000 20000 ];
GMAT MoonInertialView.ViewDirection = Luna;
GMAT MoonInertialView.ViewScaleFactor = 1;
GMAT MoonInertialView.ViewUpCoordinateSystem = MoonInertial;
```

```
GMAT MoonInertialView.ViewUpAxis = Z;
GMAT MoonInertialView.EclipticPlane = Off;
GMAT MoonInertialView.XYPlane = On;
GMAT MoonInertialView.WireFrame = Off;
GMAT MoonInertialView.Axes = On;
GMAT MoonInertialView.Grid = Off;
GMAT MoonInertialView.SunLine = Off;
GMAT MoonInertialView.UseInitialView = On;
GMAT MoonInertialView.StarCount = 7000;
GMAT MoonInertialView.EnableStars = On;
GMAT MoonInertialView.EnableConstellations = On;

Create OrbitView EarthInertialView;
GMAT EarthInertialView.SolverIterations = Current;
GMAT EarthInertialView.UpperLeft = [ 0.5310300703774792 0.1855036855036855 ];
GMAT EarthInertialView.Size = [ 0.5067178502879078 0.3525798525798526 ];
GMAT EarthInertialView.RelativeZOrder = 73;
GMAT EarthInertialView.Maximized = false;
GMAT EarthInertialView.Add = {MoonSat, Earth, Luna};
GMAT EarthInertialView.CoordinateSystem = EarthMJ2000Eq;
GMAT EarthInertialView.DrawObject = [ true true true ];
GMAT EarthInertialView.DataCollectFrequency = 1;
GMAT EarthInertialView.UpdatePlotFrequency = 50;
GMAT EarthInertialView.NumPointsToRedraw = 0;
GMAT EarthInertialView.ShowPlot = true;
GMAT EarthInertialView.MaxPlotPoints = 20000;
GMAT EarthInertialView.ShowLabels = true;
GMAT EarthInertialView.ViewPointReference = Earth;
```

```
GMAT EarthInertialView.ViewPointVector = [ 0 0 30000 ];
GMAT EarthInertialView.ViewDirection = Earth;
GMAT EarthInertialView.ViewScaleFactor = 1;
GMAT EarthInertialView.ViewUpCoordinateSystem = EarthMJ2000Eq;
GMAT EarthInertialView.ViewUpAxis = Z;
GMAT EarthInertialView.EclipticPlane = Off;
GMAT EarthInertialView.XYPlane = On;
GMAT EarthInertialView.WireFrame = Off;
GMAT EarthInertialView.Axes = On;
GMAT EarthInertialView.Grid = Off;
GMAT EarthInertialView.SunLine = Off;
GMAT EarthInertialView.UseInitialView = On;
GMAT EarthInertialView.StarCount = 7000;
GMAT EarthInertialView.EnableStars = On;
GMAT EarthInertialView.EnableConstellations = On;

Create XYPlot XYPlot1;
GMAT XYPlot1.SolverIterations = Current;
GMAT XYPlot1.UpperLeft = [ -0.06845809341010876 0.2862407862407862 ];
GMAT XYPlot1.Size = [ 1.162507997440819 0.769041769041769 ];
GMAT XYPlot1.RelativeZOrder = 15;
GMAT XYPlot1.Maximized = false;
GMAT XYPlot1.XVariable = MoonSat.ElapsedDays;
GMAT XYPlot1.YVariables = {MoonSat.Luna.Altitude};
GMAT XYPlot1.ShowGrid = true;
GMAT XYPlot1.ShowPlot = true;

%----------------------------------------
```

```
%---------- Arrays, Variables, Strings
%----------------------------------------
Create Variable RAAN AOP;
GMAT RAAN = 0;
GMAT AOP = 0;
```

```
%----------------------------------------
%---------- Mission Sequence
%----------------------------------------

BeginMissionSequence;
Target 'FineLunarTarget' DefaultDC {SolveMode = Solve, ExitMode = DiscardAndContinue,
ShowProgressWindow = true};
   Vary 'Vary RAAN' DefaultDC(MoonSat.RAAN = 200, {Perturbation = 0.0001, Lower = -
9.99e300, Upper = 9.99e300, MaxStep = 2, AdditiveScaleFactor = 0.0,
MultiplicativeScaleFactor = 1.0});
```

```
   Vary 'Vary AOP' DefaultDC(MoonSat.AOP = 0, {Perturbation = 0.0001, Lower = -9.99e300,
Upper = 9.99e300, MaxStep = 2, AdditiveScaleFactor = 0.0, MultiplicativeScaleFactor = 1.0});

   Vary 'Vary TOI' DefaultDC(TOI.Element1 = 1.337030302898004, {Perturbation = 0.0001,
Lower = -9.99e300, Upper = 9.99e300, MaxStep = 0.2, AdditiveScaleFactor = 0.0,
MultiplicativeScaleFactor = 1.0});

   Maneuver 'Apply TOI' TOI(MoonSat);

   Propagate 'Prop To Moon' EarthMoonProp(MoonSat) {MoonSat.Luna.Periapsis, OrbitColor =
[33 222 37]};

   Achieve 'Achieve BdotR' DefaultDC(MoonSat.MoonMJ2000EQ.BdotR = 5090, {Tolerance =
0.1});

   Achieve 'Achieve BdotT' DefaultDC(MoonSat.MoonMJ2000EQ.BdotT = 0, {Tolerance =
0.1});

EndTarget;  % For targeter DefaultDC

Target 'CircularOrbitAroundMoon' DefaultDC {SolveMode = Solve, ExitMode =
DiscardAndContinue, ShowProgressWindow = true};

   Vary 'VaryMOI' DefaultDC(MOI.Element1 = -1, {Perturbation = 0.1, Lower = -9.9e300,
Upper = 9.9e300, MaxStep = 0.2, AdditiveScaleFactor = 0.0, MultiplicativeScaleFactor = 1.0});

   Maneuver 'ApplyMOI' MOI(MoonSat);

   Propagate 'PropToApoapsis' NearMoonProp(MoonSat) {MoonSat.Luna.Apoapsis, OrbitColor
= [0 0 255]};

   Achieve 'AchieveRadApoapsis' DefaultDC(MoonSat.Luna.RadApo = 1837, {Tolerance = 2});

   Achieve 'AchieveEcc' DefaultDC(MoonSat.Luna.ECC = 0, {Tolerance = 0.001});

EndTarget;  % For targeter DefaultDC

Propagate 'LMO' NearMoonProp(MoonSat) {MoonSat.ElapsedDays = 1, OrbitColor = [255 255
0]};
```

Appendix C. MATLAB and GMAT code

https://drive.google.com/drive/folders/1P8DQBbN-rm1elbiPrUkJHJ_Y49jB5vdM