8-2014

Trajectory Optimization for a Misson to the Trojan Asteroids

Shivaji Senapati Gadsing
Western Michigan University, sgadsing08@gmail.com

TRAJECTORY OPTIMIZATION FOR A MISSON TO THE TROJAN ASTEROIDS

by

Shivaji Senapati Gadsing

A thesis submitted to the Graduate College
in partial fulfillment of the requirements
for the Degree of Master of Science
Mechanical and Aerospace Engineering
Western Michigan University
August 2014

Thesis Committee:

Jennifer Hudson, Ph.D., Chair
James Kamman, Ph.D.
Kapseong Ro, Ph.D.
Christopher Cho, Ph.D.

TRAJECTORY OPTIMIZATION FOR A MISSON TO THE TROJAN ASTEROIDS

Shivaji Senapati Gadsing, M.S.

Western Michigan University, 2014

The problem of finding a minimum-fuel trajectory for a mission to the Jovian Trojan asteroids is considered. The problem is formulated as a modified traveling salesman problem. Two different types of algorithms such as an exhaustive search algorithm and a serial rendezvous search algorithm are developed. The General Mission Analysis Tool (GMAT) is employed for finding optimum trajectories with minimal fuel consumption. The selection of a minimum-fuel mission trajectory, and the associated target asteroids, will be a key factor in determining feasibility and scientific value of a Trojan tour and rendezvous mission.

The transfer trajectory followed by a spacecraft between two orbital states can be calculated by solving Lambert's problem. Matlab language is extensively used to establish the intercommunicating interface between GMAT software and Lambert's solution. The results achieved by solving Lambert's problem and dynamic programming algorithm in Matlab are directly passed to GMAT software for higher-fidelity trajectory optimization and visualization of the trajectory. The comparison between the results obtained is verified by minimum delta-v criteria.

In this thesis several cases of asteroid selection are taken into consideration. An exhaustive search approach, which considers every possible permutation of the order of asteroid visits, is employed up to the practical limit of eight asteroids. A larger number of Trojan asteroids sets require more efficient methods; a serial rendezvous search is employed for larger sets. Also a range of mission dates and transfer times are considered.

ACKNOWLEDGMENTS

It is my pleasure to thank all those who made this thesis possible. I want to thank the Department of Mechanical and Aerospace Engineering for providing all necessary facilities to complete this work. I owe my deepest gratitude and thank to my advisor, Dr. Jennifer Hudson whose guidance; support and encouragement enabled me to develop an understanding of this subject. Her valuable advice during my MS program helped me in maintaining my focus and reaching my potential. I am also thankful to her for providing me with all the resources necessary for this research. I would like to thank my advisory committee members Dr. James Kamman, Dr. Kapseong Ro and Dr. Christopher Cho for their support and insightful suggestions. Their kind support and guidance have been of great value in this study.

I also want to thank Mr. Koorosh Naghshineh for his suggestions and help during my work. We are really glad to have a person like him in the Mechanical and Aerospace Engineering department, helping all the students. I want to thank all my friends for their constant support and encouragement, which has helped me a lot in achieving my goals. My deepest gratitude also goes to all those who directly and indirectly helped me in making this work possible.

I really fall short of words, while expressing my gratitude towards Lord Shankar and my beloved parents Mr. Senapati Rajaram Gadsing and Mrs. Suman Senapati Gadsing, whose blessings have helped me achieving my goals till date.

Shivaji Senapati Gadsing

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

The Jovian Trojan asteroids share Jupiter's orbit. They form two groups, 60° ahead of and behind Jupiter, in locations known as the L4 and L5 Lagrange points. The Trojan asteroids are a separate dynamic group from the well-known main belt asteroids, which are approximately situated between the orbits of Mars and Jupiter in the solar system. Planetary scientists believe that the Trojan asteroids hold important clues to the origin and evolution of the solar system. The physical properties of the Trojan asteroids could support one of the two competing theories on the solar system's origin and evolution; their material composition could indicate whether these asteroids originated near Jupiter's orbit or migrated from the outer solar system (i.e. the "Nice model").

The Trojan asteroids have never been visited by spacecraft, so our scientific understanding of them is far from complete. The Trojan asteroids tend to have low albedo with no sign of presence of water in contrast to main belt asteroids. In depth study of interior and exterior characteristics of Trojan asteroids will contribute to knowledge regarding the weathering and evolution of these Trojan asteroids. As discussed by Barbee et al. [1] in regard to main belt asteroids, a spacecraft mission that visits several asteroids is most desirable.

Target and trajectory selection are key driving factors in determining feasibility and scientific value of a mission to the Trojan asteroids. The Lagrange points of Jupiter's orbit are challenging interplanetary targets. Preliminary mission concept studies have estimated a 10-year flight time and a total delta-V of 3.9 km/s for the interplanetary trajectory [2]. Such mission designs require a long path with extensive mission duration, which subsequently necessitates large fuel mass and limits instrument-carrying capacity.

Trajectory selection for a Trojan tour is a complex global optimization problem, involving a large set of potential targets and their time-varying relative states. Many methods exist to solve global trajectory optimization problems [3, 4, and 5]. This thesis describes a new method to evaluate the large set of potential Trojan mission trajectories; the technique uses the open-source General Mission Analysis Tool (GMAT) [6] and dynamic programming approach for finding the optimum trajectory with minimal fuel consumption.

In this study, the intra-Trojan segment of the mission will be under consideration. That is, this analysis considers only the portion of the trajectory that occurs between asteroids after the spacecraft has arrived at the Trojan cloud. The selection of a minimum-fuel trajectory, and the associated target asteroids, will be a key factor in determining feasibility and scientific value of a Trojan tour and rendezvous mission. The objective of the research mentioned here is to design an optimal trajectory which will tour a significant number of Trojan asteroids while maximizing spacecraft payload.

Trojan Asteroids

Lagrange points (Libration Points) are five points in a three-body orbital configuration where a small body can maintain constant position with respect to two larger bodies. The gravitational force exerted by larger bodies cancels the centripetal force of the system's rotation. A spacecraft at one of the Sun-Jupiter Lagrange points follows a near-circular orbit around the Sun with the same period as that of Jupiter.

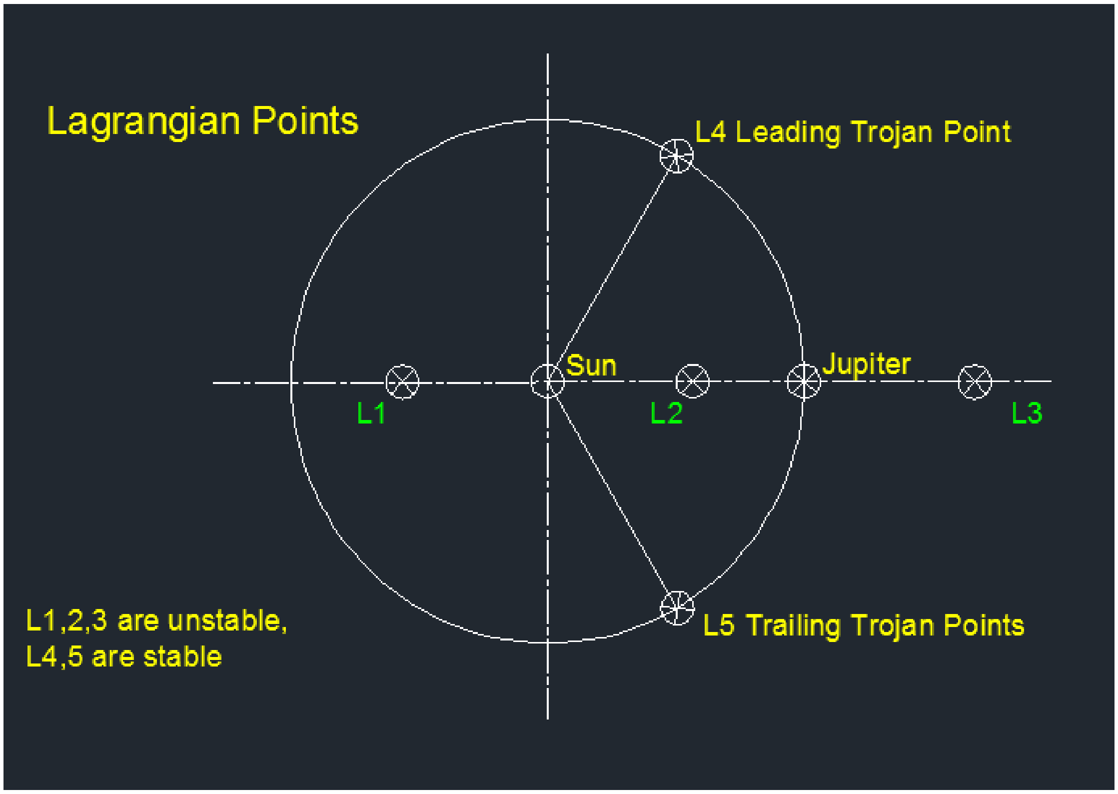


Figure 1. Location of L1, 2,3,4,5 Trojan asteroids.

The Trojan asteroids share Jupiter's orbit and occupy the L4 and L5 Lagrange points. Asteroids in the leading cloud are named to represent Greek warriors; those in the trailing cloud represent defenders of Troy. The L4 cloud contains a larger number of asteroids: approximately 1.34 times as many as in the L5 cloud [7]. Recent studies from NASA's Wide-field Infrared Survey Explorer have shown that the Trojan asteroids are composed of predominantly dark and reddish rock with non-reflecting surfaces [8].

There are two theories on the formation of Trojan asteroids. One theory postulates that they were seized into libration points during the formation of Jupiter. The second theory hypothesizes that they were pulled into their current location as a result of planetary migration. The high inclination of many Trojan asteroids conflicts with the first theory. The authentication of either of the concepts will require a dedicated space mission [9].

Figure 2 shows orbital element data for 3,004 Trojan asteroids in the L4 and L5 groups were obtained from the JPL Small-Body Database [10]. The L4 group asteroids are represented by large cluster on left side while the L5 group asteroids are sparse on the right side.

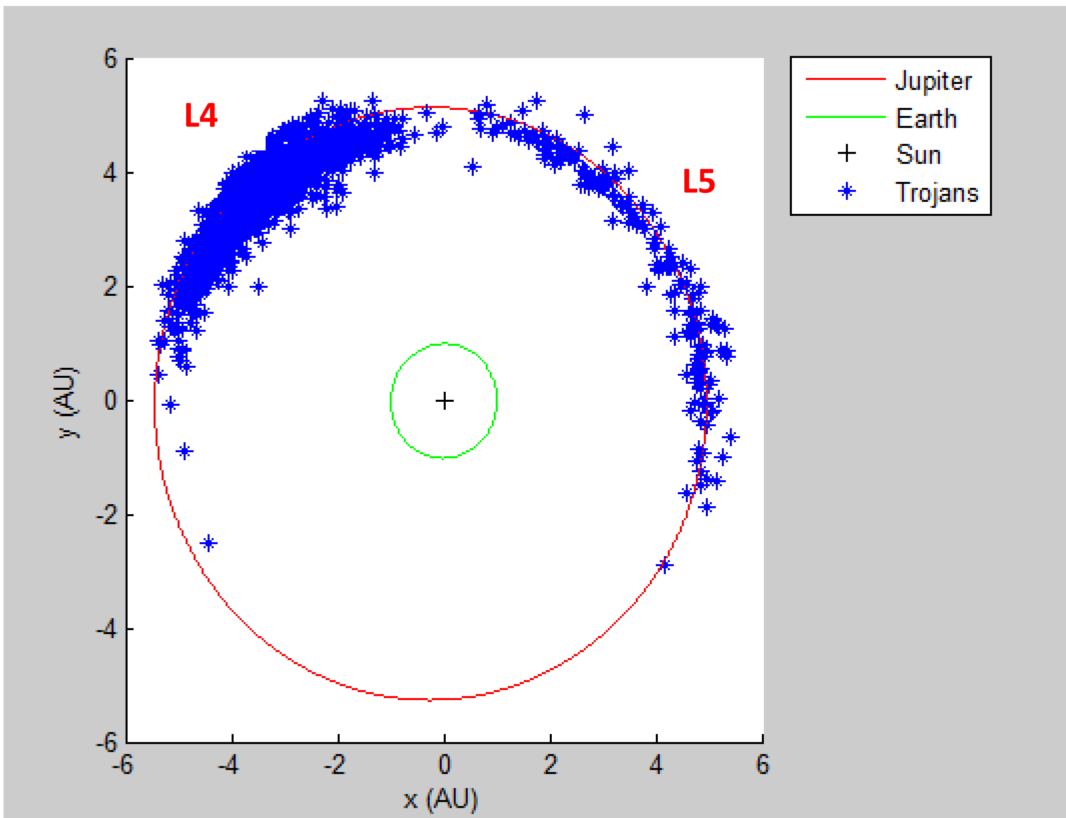


Figure 2. Locations of 3,004 Trojan asteroids.

Outstanding Issues and Science Questions to Address

The Trojan asteroids are central to a number of major questions in planetary science [11]. The NASA 2013 decadal survey recommends a Trojan Tour and Rendezvous mission as one of five candidates for New Frontiers Mission 4 [12]. The mission recommends specimen material from Jovian asteroid region. These acquired experimental facts will provide insightful observations and results for space weathering and process affecting these Trojan asteroids.

Major scientific questions about the Trojan asteroids are as follows:-

1) Source of origin of Jovian Trojan asteroids.
2) Physical property relationship for indication between Trojan asteroids and solar nebula region.

To answer the above supreme questions, Trojan asteroids must be characterized and placed in context with other primitive bodies and the outer solar system [1]. Most of the knowledge about these asteroids is solely formed by Earth based observations. In order to leverage our knowledge, an exclusively motivated and dedicated mission is mandatory for exploration of Trojan asteroids which will profoundly answer many questions.

M.A. Barucci and D.P. Cruikshank [13] described various properties of Trojan asteroids such as:

1). Rotational Properties: Shape and angular momentum were acquired during solar system accretion. The characterization of these properties can provide important clues to the history of the Trojans. Light curve observations represent the basic tool for determining the rotational properties of asteroids, allowing for the determination of the rotation rate and angular momentum direction, as well as an approximation of body shape. A major observational program is currently under way to systematically explore the rotational properties of Trojans.

2). Lightcurve Amplitudes: The amplitude of a lightcurve is an indicator of an asteroid's elongation. The amplitude, however, varies considerably depending on the unknown aspect angle under which the observations are made, with the amplitude being largest with an equatorial aspect and smallest with a polar aspect.

Orbital Maneuvers

Interplanetary mission propulsion

An interplanetary mission could use different types of propulsion system such as solar electric propulsion (Low Thrust) or chemical propulsion (High Thrust). For interplanetary mission, the type of propulsion system has a great impact on trajectory time which in turn affects mass and mission operation costs. Furthermore, taking into account the power requirements, we can use different propulsion systems for different phases in the spacecraft trajectory.

Solar electric propulsion is a form ionic propulsion in which the power for the ion engine is supplied by the solar cell panels. Solar electric propulsion uses the principle of magnetism and electricity for driving the spacecraft forward. Solar electric propulsion can be observed as best solution for interplanetary mission as they require very high change in velocity resulting into long mission duration. Also its higher efficiency will significantly reduce the fuel costs over the long mission timespan. Whereas chemical propulsion is the propulsion in which thrust is supplied by the product of a chemical reaction. The phenomenon of chemical propulsion utilizes burning a fuel, very often chemical propulsion causes reaction of H and O2 to give water. Solar electric propulsion is used in recent missions like 'Dawn' and 'Hayabusa' for trajectory control and corrections, along with rendezvous and orbit insertion whereas 'NEAR' mission utilized principle of chemical propulsion.

In this thesis study, we chose to analyze the inter-Trojan portion of the mission assuming chemical propulsion. Such kind of mission could also use hybrid propulsion system, with ion propulsion for the flight from Earth escape until arrival at L4, and then chemical propulsion for maneuvers within the L4 group. The working of hybrid propulsion utilizes principle of both chemical and electric propulsion.

Impulsive orbital transfers

M. Vasile and M. Locatelli [14] describe a simple method to find the best launch date and time of flight to transfer a spacecraft from one celestial body (such as Earth) to another one (such as an asteroid).

As described in Reference [15], an impulsive orbital transfer is simply the firing of on-board rocket motors to produce a change in the magnitude and direction of the velocity vector instantaneously. During an impulsive maneuver, only the velocity parameter gets changed. Application of an impulsive orbital transfer results in a change delta-v in the velocity of the spacecraft. Also propellant mass m is always related to change in velocity delta-v and is given by the equation.

$$\frac{m}{m} = 1 - e^{-\frac{delta-v}{Isp*g0}} \quad ----(1)$$

Where m is the mass of the spacecraft before the burn, go is the sea-level standard acceleration of gravity, and Isp is the specific impulse of the propulsion system.

For an impulsive orbital transfer the spacecraft engine is turned on for a very short period of time but with a powerful thrust. The spacecraft then executes a coast arc along the resulting Keplerian trajectory. Given the flight time from the first to the second body, the transfer orbit can be computed from the equation of motion. The solution of Lambert's problem provides the velocities at the beginning and at the end of the transfer arc. In this thesis when a spacecraft reaches one of the asteroids, then an impulsive orbital transfer will take place in order to reach the next asteroid. This trajectory transfer will take the results obtained from Lambert's problem depending upon the position of the two asteroids.

Lambert's problem

Given two points in space and the time of flight between them, the trajectory followed by the spacecraft between the two points can be calculated by solving Lambert's problem. As shown in Figure 3, the objective is to find the trajectory $r(t)$ such that a spacecraft transfers from point P1 to point P2 in fixed time Δt. The resulting trajectory is called a Lambert's arc. According to Lambert's theorem "The transfer time Δt from P1 to P2 is independent of the orbit's eccentricity and depends only on the sum r1 + r2 of the magnitudes of the position vectors, the semi major axis a, and the length c of the chord joining P1 and P2". [15]

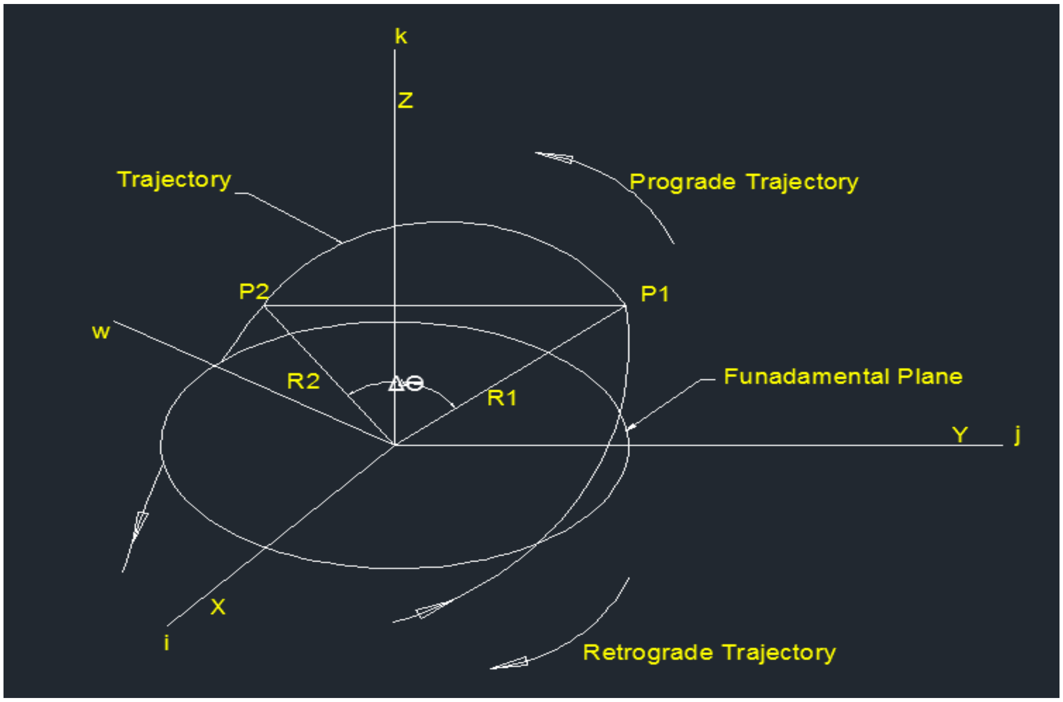


Figure 3. Trajectory representation.

Lambert's problem is a boundary value problem with dynamics governed by the fundamental equation of relative two-body motion,

$$\ddot{\vec{r}} = -\frac{\mu}{r^3} * \vec{r} \quad ------(2)$$

The boundary conditions are

$$r(t_1) = r_1 \quad ------(3)$$

$$r(t_2) = r_2 \quad ------(4)$$

In this analysis, we solve Lambert's problem for transfers between pairs of Trojan asteroids. P1 and P2 represent the positions of two asteroids at the initial and final time, respectively, of the transfer. Lambert's problems usually have multiple solutions, but we can introduce further constraints to reduce the number of solutions to one.

The spacecraft can follow either the prograde trajectory or the retrograde trajectory, as shown in Figure 4. The prograde trajectory follows the direction of motion of the asteroids about the sun; the retrograde trajectory proceeds in the opposite direction. The inclination of the orbit is defined for these cases:

1) Prograde Trajectories ($0 < i < 90°$)

2) Retrograde Trajectories ($90° < i < 180°$)

In this analysis, both the prograde and retrograde trajectories were considered for each trajectory segment, and the option with the lowest fuel cost was selected. In most cases, the prograde trajectory was the most fuel-efficient.

Formulation of Trajectory Optimization Problems

Giovanni et al. [16] describe the essential problem in trajectory optimization for interplanetary trajectories. The boundary conditions consist of launching a spacecraft from an astronomical body (such as Earth) along a trajectory which will meet the desired destination. The objectives of trajectory optimization mission can be minimal cost, minimal mission duration or minimal fuel expenditure. Hence, trajectory optimization problems for space missions can be categorized by large search spaces.

Spacecraft trajectory optimization typically falls under the category of constrained, non-linear optimization.

1). Cost Function: Optimization can be performed using the following types of objective functions.

1). Maximization of spacecraft mass.

2). Minimization of total tour time.

3). Minimization of fuel consumption.

2). Variables: The independent variables associated with a spacecraft are mass, position and velocity. A crucial independent variable is the thrust vector which can be treated as a function of time or position. Other potential independent variables will be the power of spacecraft and specific impulse of thruster. In this thesis, constant specific impulse is assumed.

3). Optimization: Jon A. Sims et al. [17] describe that the optimizer adjusts the independent variables to satisfy all of the bounds and constraints while simultaneously optimizing the cost function. If the optimizer can find a set of variables that satisfies the bounds and constraints, the trajectory defined by those variables is said to be feasible.

Various types of algorithm for optimization are as follows:-

1). Global Optimization Algorithm

2). SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization.

3). SAGES Algorithm (Self-Adaptive Gaussian Evolution Strategies).

4). Multi agent Collaborative Search

5). Differential Evolution

6). Genetic Algorithm

Dynamic programming algorithm

Dynamic programming is a method for efficiently solving problems that are composed of overlapping sub problems. The basic structure of this programming approach was used in the trajectory optimization methods described in this paper.

Principle of optimality defined by Richard Bellman as an optimal path has the property that whatever is the initial state and initial decision over the given period of time, the control variables chosen for the remaining period must be optimal for remaining problem [18].

Dynamic programming relies on Bellman's Principle of Optimality [18] giving necessary and sufficient conditions for a solution to be locally optimal. Trajectory optimization methods for dynamic programming are classified into direct and indirect methods. Direct methods are more robust. Even if a first guess is poor, these methods typically assure convergence. Indirect methods guarantee accuracy but require a good first guess. Iterations of dynamic programming produced through backward propagation of the Bellman equation often result in improved trajectories and reduction in the mission cost.

Camilla Colombo et al. [19] demonstrated dynamic programming applied to the design of rendezvous and fly-by trajectories to various objects (Asteroids and Comets). In these techniques, high dimensional problem characteristics of low thrust trajectory optimization can be reduced into a series of small dimensional problem.

In this research a dynamic programming algorithm is developed for the problem of finding a minimum-fuel, high-thrust trajectory for a mission to several Jovian Trojan asteroids. The problem is formulated as a modified traveling salesman problem.

The travelling salesman problem is one of most studied topics in computer science and operations research. The travelling salesman problem is a classical problem in optimization and graph theory. It poses the question: given a list of cities and the distance between them, what is the minimum–distance route that passes through each city and returns to starting location? As the travelling salesman problem is of high computational complexity, the only one way to fully solve it is to evaluate every possible path; there are a set of feasible solutions for travelling salesman problem and is given as (n-1)! Solutions. The application of travelling salesman problem and linkages with other problems can found in drilling of printed circuit board, overhauling gas turbine engine, X-ray crystallography, computer wiring, vehicle routing, and order-picking problem in warehouses. The travelling salesman problem is of the type NP hard, a class of decision making problem.

The abbreviation NP refers to ‘Nondeterministic polynomial’. There are several ways of solving travelling salesman problem such as graph algorithm, heuristic and approximation algorithms. If any algorithm for travelling salesman problem increases exponentially with increase in the number of targets, results will be undesirable.

The Trojan asteroid mission trajectory problem is a modified travelling salesman problem, which can be solved by the Branch and Bound, Heuristics, and Direct Graph algorithms. In this paper, in the exhaustive search method, we employ the travelling salesman problem solution by Richard Bellman’s Direct Graph algorithm [20].

The mission tour is a one-way trip, originating at an initial state x that corresponds to the position and velocity of one of the Trojan asteroids. Out of a set of n asteroids in the L4 group, the objective is to find the minimum-fuel trajectory that passes within 6000 km of m asteroids, where $m = n$ (Case 1) or $m < n$ (Case 2).

At the i^{th} flyby of the optimal tour, with k flybys remaining, where m = k – i, we define

$f(i; j_1, j_2,..., j_k)$ = Cost of a minimum-fuel trajectory that passes once and only once by each of the remaining k asteroids $j_1, j_2,..., j_k$. …… (5)

We define $d_{i,j}$ to be the fuel cost between the i^{th} and j^{th} flybys. An iterative procedure is initiated with

$$f(i; j) = d_{i,j} + d_{jx}. \quad ………… (6)$$

The iterations are repeated until $f(x; j_1, j_2,..., j_m)$ is obtained.

The problem is asymmetric (a trajectory from asteroid A to asteroid B may have a different fuel cost than a trajectory from B to A) and fuel costs are time-varying based on the relative states of the target asteroids at the time of flight. As such, a trajectory optimization tool (MATLAB or GMAT) is used to calculate $d_{i,j}$ for each trajectory segment.

In this analysis we formulate the travelling salesman problem with each asteroid as a nodal point. Using n = 4 asteroids in this preliminary analysis, the possible permutations of tour order are 4! = 24 possible paths. In our analysis, out of these 24 possible paths, the optimum trajectory will be that with minimum total delta-v.

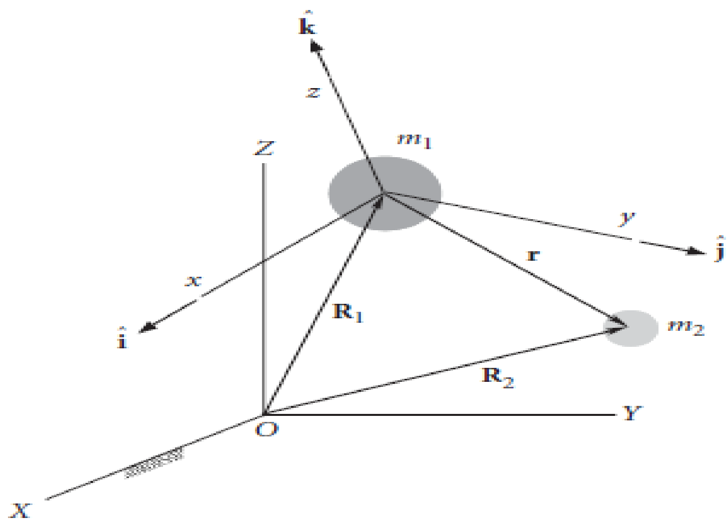
Problem Statement

This thesis focuses on the Trojan mission trajectory design problem. Spacecraft trajectory design can be treated as a global trajectory optimization problem. Each space mission requires finding an optimal trajectory while taking into account the constraints such as fuel and time needed for mission accomplishment.

This thesis develops computational algorithms for the design of an interplanetary trajectory transfer. During this study the objective function will be to minimize the propellant mass required to perform trajectory transfer. Moreover from equation [Section II c ii-1], propellant mass increases exponentially with delta-v, so this analysis will focus on minimum delta-v solutions.

II. MATHEMATICAL MODEL OF TRAJECTORY OPTIMIZATION

We will define the problem as a restricted two-body problem. A spacecraft of negligible mass orbits the Sun. In general the independent variable under consideration is time t. We are trying to find minimum delta-v for a series of asteroid flybys. The dynamics of system can be described by the equation of motion.



Let $\vec{r}$ be the position vector of m_2 relative to m_1 then

$\vec{r} = \vec{R}_2 - \vec{R}_1$

$\vec{r} = \ x_2 - x_1)\hat{\imath} + (y_2 - y_1)\hat{\jmath} + (z_2 - z_1)\hat{k}$

$\vec{r} = x\hat{\imath} + y\hat{\jmath} + z\hat{k}$

Where x = $x_2 - x_1)$

y = $y_2 - y_1)$

z = $z_2 - z_1)$

$r = \sqrt{x^2 + y^2 + z^2}$

Also velocity $\vec{v} = \dot{\vec{r}} = (\dot{x})\hat{\imath} + (\dot{y})\hat{\jmath} + (\dot{z})\hat{k}$

The components of the state equation are given by

$$\overrightarrow{y} = \begin{bmatrix} \vec{r} \\ \vec{v} \end{bmatrix}$$

This is given by the above result as

$$\vec{y} = [x \;\; y \;\; z \;\; \dot{x} \;\; \dot{y} \;\; \dot{z}]$$

The time derivative of this state vector will give rise to equation of motion as

$$\dot{\vec{y}} = \begin{bmatrix} \vec{v} \\ \ddot{\vec{r}} \end{bmatrix} \qquad ------ \;\; 7)$$

Where $\ddot{\vec{r}} = -\frac{\mu}{r^3} * \vec{r}$

which can be given as

$$\dot{\vec{y}} = [\dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \ddot{x} \;\; \ddot{y} \;\; \ddot{z}]$$

The initial condition of the two-body boundary value problem is that spacecraft will have position and velocity equal to the first asteroid under consideration. The final condition is the target asteroid position within 6000 km.

III. METHODS

In this thesis, to develop the methods and software algorithms for solving the global trajectory optimization problem, a subset of Trojan asteroids was considered. Within the L4 group, 16 asteroids were selected; 12 of these are among the largest, earliest-discovered, and best-characterized Trojans, the remaining four were selected based on low orbital eccentricity and inclination.

Table 1: Trojan Asteroid List

588 Achilles	911 Agamemnon	1404 Ajax	624 Hektor	659Nestor
1143 Odysseus	1437 Diomedes	1583 Antioclus	1647 Meneclaus	1749 Telamon
1868 Thersites	1869 Philoctetes	2146 Sretor	2148 Epeios	2260 Neoptolemus
2456Palamedes	2759 Idomenus	2797 Teucer	2920 Automedon	3063 Makhaon
89913 (2002 EC24)	228108 (2008 SU277) – M	263795 (2008 QP 41) – S	316146 (2009 SV347) – Si	

To simplify the problem, only the inter-Trojan segments of the trajectory were considered. The full Trojan mission trajectory will likely require gravity assists at Jupiter and other planets. These trajectory segments, which depend largely on launch window, will not be considered. Many of the Trojans orbits have inclinations and eccentricities that differ significantly from Jupiter's orbit. Thus, the relative positions and velocities of asteroids within each cloud are highly time dependent. In this study, the start time will be varied and transfer times will be treated as free variables to optimize these time-dependent relationships.

Exhaustive Search

Exhaustive search is a method of evaluating all possible permutation results for a given problem. In this preliminary analysis, we introduced n = 4 asteroids with the possible permutations of tour order are 4! = 24 possible paths. In our analysis, out of these 24 possible paths, the optimum trajectory will be that with minimum total delta-v.

Similarly when n = 8 asteroids, the possible permutation of tour order are 8! = 40,320 possible paths. Out of these possible paths, the optimum trajectory will be that with minimum total delta-v. In this way fixing the initial asteroid, the remaining n = 7 asteroids, the possible permutation of tour order are 7! = 5,040 possible paths and so on repeating the above procedure. In this way the sequence for optimum trajectory transfer will be premeditated.

Serial Rendezvous Search

Serial Asteroid Rendezvous Problem

Brent Barbee et al. [1] demonstrated the Serial Asteroid Rendezvous method to find an ordered set of asteroids with spacecraft departure from Earth and rendezvous with each asteroid. In their problem formulation, a spacecraft may stay for some time on each asteroid while exploring the properties of asteroids through sample material collected. This spacecraft will follow an optimum trajectory to return to Earth with all samples collected.

In our research a set of 8 asteroids are selected from various Trojan asteroids. The year of launch from Earth is kept between 2013 and 2017. A constraint of mission duration to be less than 10 year is maintained. The time of flight between two asteroids is varied between 50, 100, and 150 days.

In this research there is no limitation on spacecraft mass, propulsion system, or Earth departure velocity. The goal of this mission is to minimize the value of delta-v required for the mission resulting in an optimal trajectory transfer tour.

Solution Methodology:

The Series method Algorithm is used to find feasible itineraries for a serial Trojan asteroid rendezvous problem. The solution of this method minimizes the value of delta-v for the mission by selecting each asteroid in the itinerary according to minimum rendezvous delta-v criteria.

First, an asteroid is selected as the starting point. Following the method in [1], all possible trajectories from the first asteroid to remaining 'X' asteroids are computed. Departure date and time of flight are varied and a grid of possible solutions is formed (Figure 4). One of 'X' asteroids which will give minimum delta-v is selected and the trajectory is propagated to that point. After selection of first asteroid, 'X-1' asteroids are still remaining in population from which the second rendezvous target will be selected. The new grid of trajectory points will be achieved by letting transfer time from the current asteroid to the next asteroid vary in steps of 50, 100, and 150 days and also changing starting date 60 and 120 days. Thus each trajectory grid point is associated with varying individual transfer time and date. A Lambert's targeting algorithm is used for all the trajectory calculation.

This operation is performed for all the remaining ‘X-1’ asteroids in the population, and the asteroid which will give minimum delta-v is selected for second asteroid rendezvous. The above operation is repeated continuously to find remaining third, fourth and so on till last asteroid.

The complete asteroid tour itinerary is not necessarily an optimal trajectory transfer tour. Solutions with lower total delta-v may be found. For example, choosing a first target with a slightly higher delta-v may enable subsequent trajectory segments with much lower delta-v’s. However, the series method does allow low delta-v trajectories to be found from a large set of targets with much less computational time than the exhaustive search method.

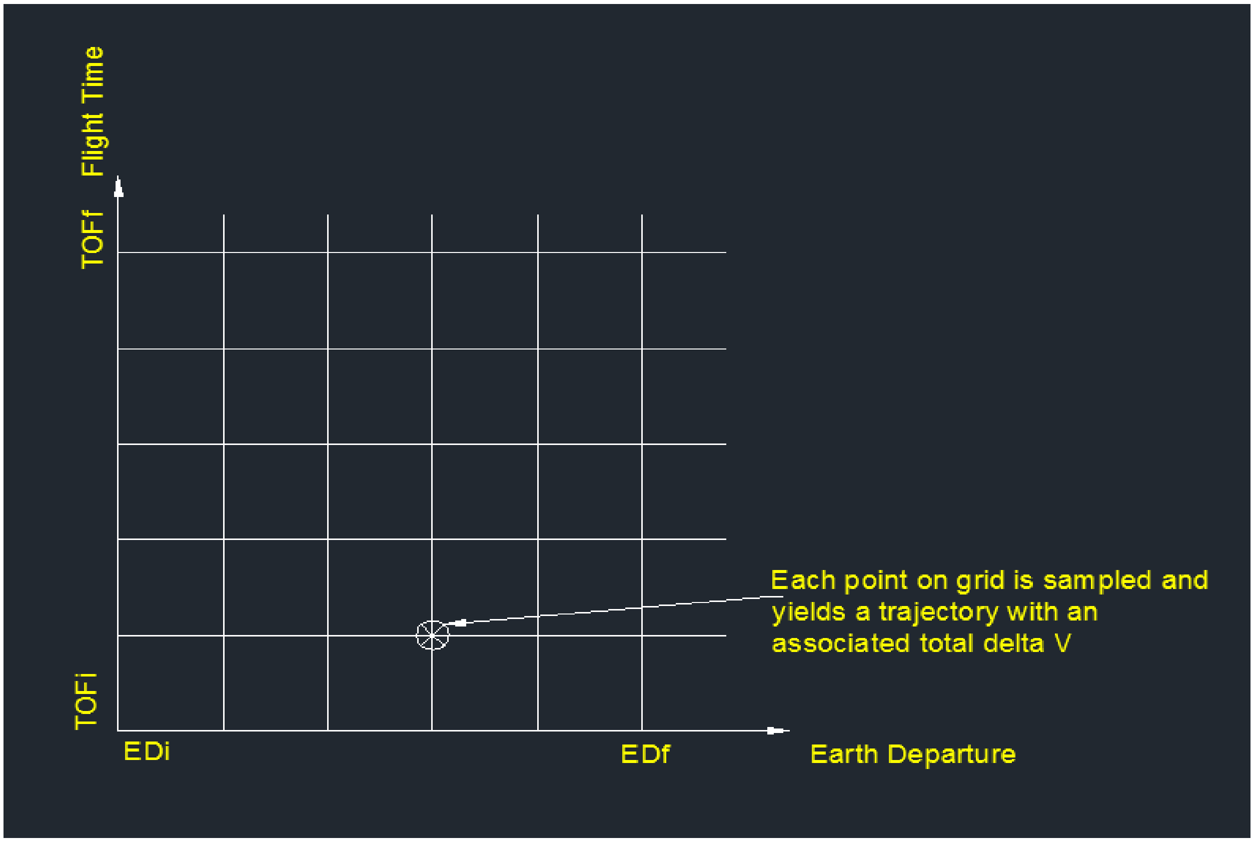


Figure 4. Earth Departure versus time of flight.

Implemented Software

Matlab

The code developed in Matlab is used for calculating the total delta-v for visiting a series of asteroids. The list of asteroids to be visited is given in the cell array ‘target’ along with its NAIF id’s obtained from NASA. The time increment between the dates is varying factor for several cases under consideration. The necessary Spice kernels to define various properties of asteroids are obtained from NASA website. [21] and which is coded into Matlab language. This program calculates all possible permutation of visiting order and the delta-v associated with each asteroids. Results of Tour_length will store the total delta-v for each path whereas dv_sols will store the impulsive burn solutions for each segment.

The function used to solve the Lambert's problem in Matlab language is as follows.

$$function[V_1\ V_2] = lambert\_asteroids(R_1, R_2,\ t, string\ )$$

Where R1, R2 Initial and final position vector (km), and V1, V2 Initial and final velocity vectors.

The trajectory obtained from Lambert's solution can be in both the retrograde and prograde direction. Matlab program solves for all possible Tour_length and Tour_order but the trajectory transfer with minimum delta-v is selected as optimal trajectory. The results obtained in Matalb are then forwarded to GMAT to run the simulation to achieve the optimal trajectory transfer between the given sets of asteroid. In the case of four asteroids, only 3 burns are under consideration for GMAT targeting.

GMAT

GMAT is an open-source tool developed by NASA and private industry that can be used to develop new space trajectory optimization and mission design technology [22]. It includes high-fidelity modeling, nonlinear constrained optimization and targeting routines, MATLAB, and GUI interfaces.

In this paper, we employ GMAT software to calculate the optimal trajectories between the Trojans within the dynamic programming algorithm. Starting at the first asteroid on each path, GMAT calculates the required impulsive burn to intercept the next asteroid at the target time, and then simulates the trajectory. The final state, which has a position corresponding to the target asteroid (within 6000 km) but different velocity, is used the initial state for the next trajectory segment. The total cost of each path is calculated as the sum of the delta-v of each segment.

To initialize the GMAT solver, Lambert's problem is solved for each pair of initial and target states. The Lambert solution gives an approximate initial velocity vector,

$$v(t+) = v(t-) + \Delta V \quad \text{- - - - - - - - (8)}$$

Where v(t-) is the spacecraft velocity prior to the impulsive burn and v(t+) is the velocity after burn. This required delta-v is then passed to GMAT to initiate the nonlinear optimizer. A high-fidelity solution for the actual delta-v of the orbit transfer is obtained from GMAT. MATLAB stores the total delta-v cost of each path permutation and selects the optimal path.

IV. RESULT CASES

Exhaustive Search: Four Asteroids

First, the exhaustive search algorithm was validated on a test set of four Trojan asteroids: 588 Achilles (Ac), 624 Hektor (H), 911 Agamemnon (Ag), and 659 Nestor (N). Two problem cases were considered. In Case 1, the trajectory was required to include only three of the four asteroids ($m < n$). In Case 2, the trajectory was required to include all four asteroids in the set ($m = n$). The mission date for this transfer is starting from 1 January 2012 with transfer time increment of 200 days between each asteroid. Hence plotted trajectory transfer consists of mission dates as 1 January 2013, 20 July 2013, 4 February 2014, and 23 August 2014 respectively.

Case 1

The MATLAB-GMAT algorithm calculated the total delta-v for a trajectory through all permutations of three of the four asteroids. The five trajectories with the smallest delta-v are shown in Table 2. Of these optimal trajectories, none include 659 Nestor.

Table 2: Five minimal delta-v trajectories through three out of the four Trojan asteroids: 588 Achilles (Ac), 624 Hektor (H), 911 Agamemnon (Ag), and 659 Nestor (N).

Trojan Tour Order	Total delta-v (km/s)
Ag-H-Ac	7.3
Ag-Ac-H	10.4
H-Ag-Ac	13.3
Ac-H-Ag	14.8

Table 2 - continued

Trojan Tour Order	Total delta-v (km/s)
Ac-Ag-H	17.7

The minimum-cost path required a total delta-v of 7.3 km/s; the highest-cost path through three of the four asteroids had a delta-v of 65.9 km/s.

Figure 5 shows the minimum delta-v trajectory simulated in GMAT.

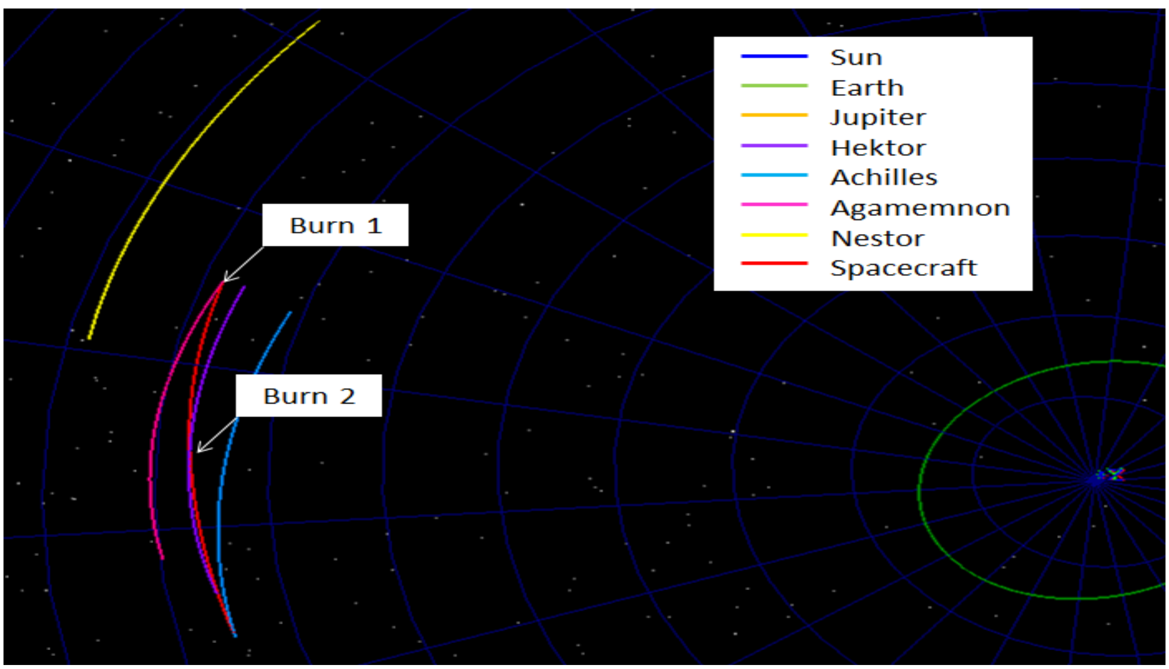


Figure 5. Minimum delta-v trajectory through three out of four Trojan asteroids.

Case 2

The MATLAB-GMAT algorithm calculated the total delta-v for each of the 24 possible paths through the four asteroids. The five trajectories with the smallest delta-v are shown in Table 3.

Table 3: Five minimal delta-v trajectories through four Trojan asteroids: 588 Achilles (Ac), 624 Hektor (H), 911 Agamemnon (Ag), and 659 Nestor (N)

Trojan Tour Order	**Total delta-v (km/s)**
H-Ac-Ag-N	34.9
Ac-H-N-Ag	35.3
N-Ac-H-Ag	38.3
Ag-N-H-Ac	38.8
Ac-N-H-Ag	41.9

The minimum-cost path required a total delta-v of 34.9 km/s; the highest-cost path through the same four asteroids had a delta-v of 93.5 km/s.

The three impulsive burns for the optimal Hector-Achilles-Agamemnon-Nestor path are given in Table 4.

Table 4: Burn vectors for the optimal path through the four Trojan asteroids.

	X (km/s)	Y (km/s)	Z (km/s)	Total |delta-v| (km/s)
Hektor to Achilles	0.594	-2.28	-5.26	5.76
Achilles to Agamemnon	-3.84	7.4	9.36	12.53
Agamemnon to Nestor	-1.21	10	13.2	16.60

Figure 6 shows the minimum delta-v trajectory. The largest trajectory change occurs on the third burn, as the spacecraft maneuvers to encounter 659 Nestor. During the 2013 timespan considered in this analysis, the position of 659 Nestor is distant from the remaining three asteroids in the set.

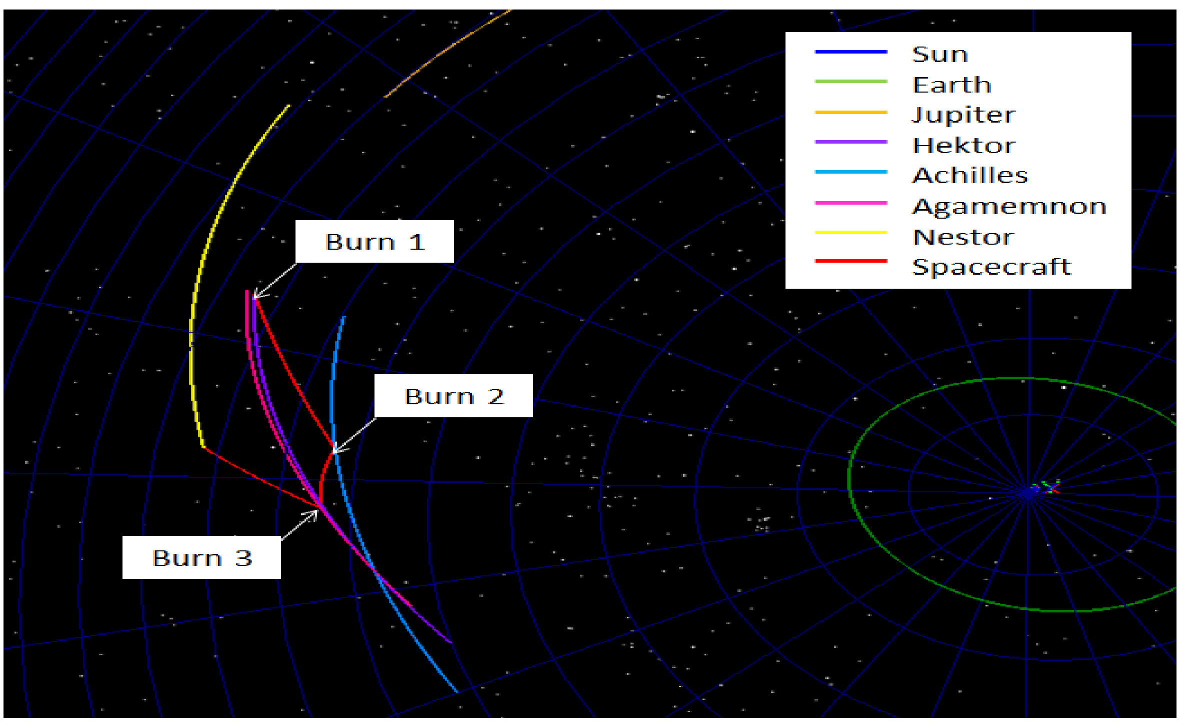


Figure 6. Minimum delta-v trajectory through four Trojan asteroids.

Exhaustive Search: Eight Asteroids

Results

The exhaustive search method was then used on a larger set of asteroids. It was found that the largest possible set that could be evaluated by the exhaustive search algorithm in a reasonable amount of time was eight asteroids. These analyses required approximately less computational time. The mission date for this transfer is starting from 1 January 2012 with transfer time increment of 200 days between each asteroid. Hence plotted trajectory transfer consists of mission dates as 1 January 2013, 20 July 2013, 4 February 2014, 23 August 2014, 11 March 2015, 27 September 2015, 14 April 2016, and 31 October 2016 respectively.

Two problem cases were considered. In Case 1, the trajectory was required to include all eight asteroids in the set ($m = n$). In Case 2, the trajectory was required to include only four out of the eight asteroids ($m < n$) through optimal path. In Case 3, the trajectory was required to include only three out of the eight asteroids ($m < n$) through optimal path.

Case 1

The MATLAB-GMAT algorithm calculated the total delta-v for each of the 1680 possible paths through the eight asteroids considering only three burns. The five trajectories with the smallest delta-v are shown in Table 5.

Table 5: Five minimal delta-v trajectories through eight Trojan asteroids: 588 Achilles (Ac), 624 Hektor (H), 3063 Makhaon (Ma), 1143 Odysseus (O), 2456 Palamedes (Pa), 1868 Thersites (Th), 2148 Epeios (Ep) , and 2759 Idomenus (Id)

Trojan Tour Order	**Total |delta-v| (km/s)**
Ac-H-Ep-O-Pa-Ma-Th-Id	13.18
O-Pa-Ep-Th-Id-Ma-H-Ac	13.60
O-Id-Ma-Ac-Ep-Th-H-Pa	14.13
Pa-Ep-H-Ac-O-Ma-Th-Id	14.46
O-Id-Pa-Th-Ep-Ma-H-Ac	14.64

The minimum-cost path required a total delta-v of 13.18 km/s; the highest-cost path through the same eight asteroids had a delta-v of 69.23 km/s.

The first Trojan tour order of seven burns with minimum delta-v out of which only first three impulsive burns for the optimal Achilles-Hector-Epeios-Odysseus path are given in Table 6.

Table 6: Burn vectors for the optimal path through eight Trojan asteroids considering only three burns.

	X (km/s)	**Y (km/s)**	**Z (km/s)**	**Total |delta-v| (km/s)**
Achilles to Hektor	-0.6176	2.2329	5.2329	5.73
Hector to Epeios	-0.1872	3.6583	-0.2592	3.68
Epeios to Odysseus	1.1738	-2.7354	-2.0478	3.63

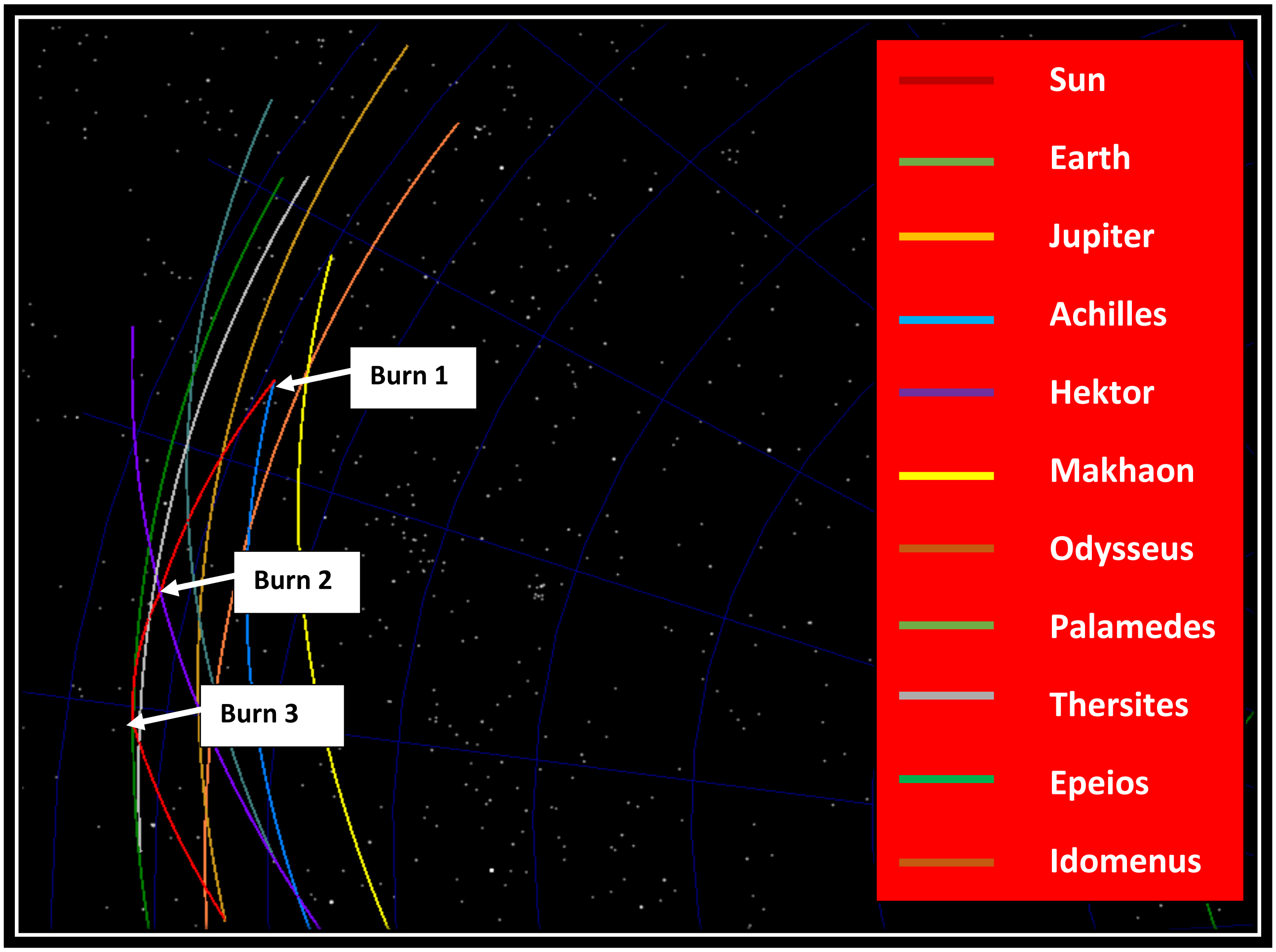


Figure 7. Minimum delta-v trajectory through eight Trojan asteroids.

Case 2

The MATLAB-GMAT algorithm calculated the total delta-v for a trajectory through all permutations of four out of the eight asteroids. The trajectories with the smallest delta-v are shown in Table 7.

Table 7: Minimal delta-v trajectories through four out of above eight Trojan asteroids: 588 Achilles (Ac), 624 Hektor (H), 2148 Epeios (Ep), and 1143 Odysseus (O).

Trojan Tour Order	Total |delta-v| (km/s)
Ac-H-Ep-O	13.18

Figure 8 shows the minimum delta-v trajectory.

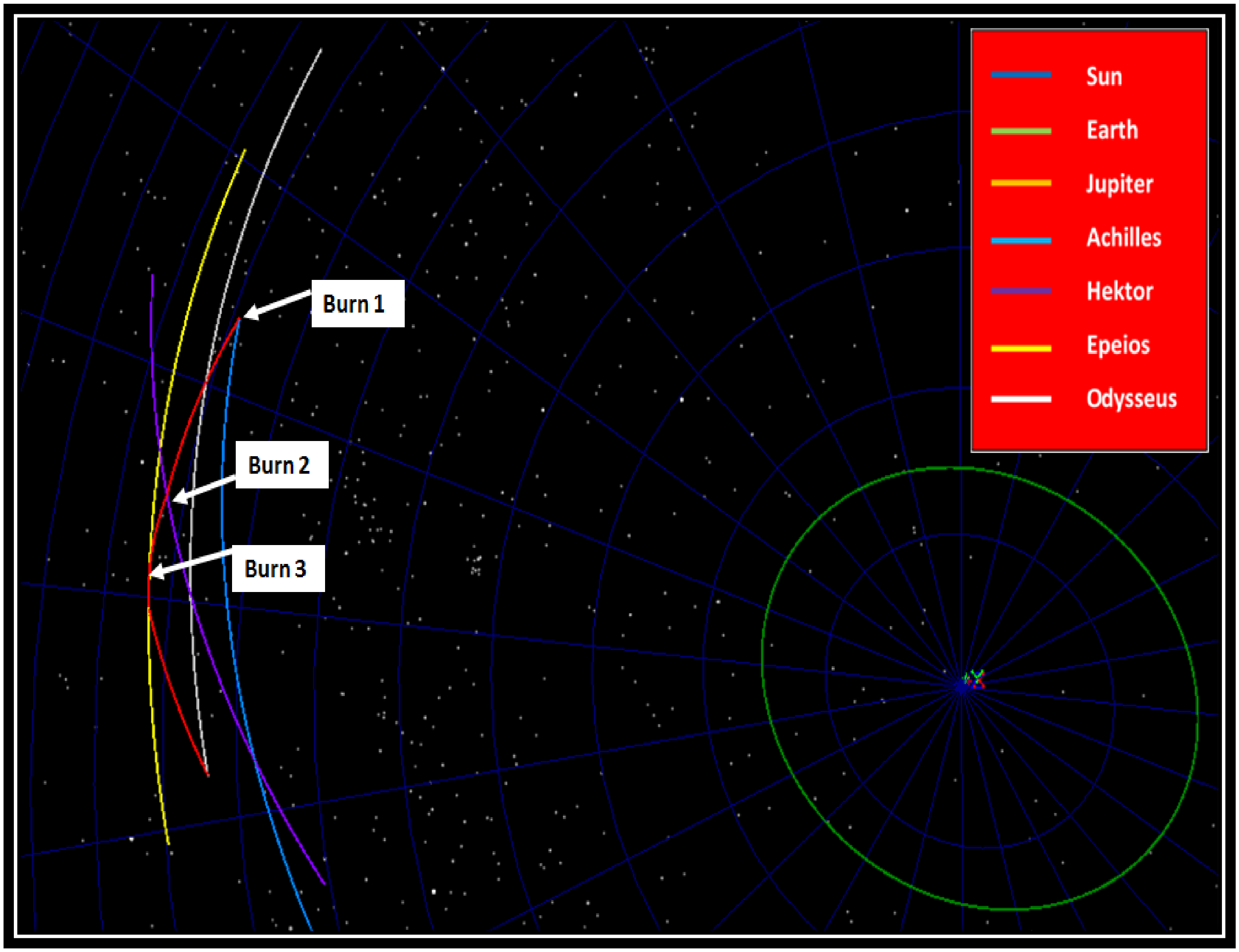


Figure 8. Minimum delta-v trajectory through four Trojan asteroids.

The minimum-cost path required a total delta-v of 13.18 km/s. The largest trajectory change occurs on the third burn, as the spacecraft maneuvers to encounter 1143 Odysseus. During the 2013 timespan considered in this analysis, the position of 1143 Odysseus is distant from the remaining three asteroids in the set.

Case 3

The MATLAB-GMAT algorithm calculated the total delta-v for a trajectory through all permutations of three out of the eight asteroids. The five trajectories with the smallest delta-v are shown in Table 8. Of these optimal trajectories, none include 1143 Odysseus.

Table 8: Five minimal delta-v trajectories through three out of above eight Trojan asteroids: 588 Achilles (Ac), 624 Hektor (H), 2148 Epeios (Ep), and 1143 Odysseus (O) is as follows.

Trojan Tour Order	**Total |delta-v| (km/s)**
Ac-H-Ep	9.47
Ep-H-Ac	16.59
H-Ep-Ac	22.78
Ep-Ac-H	23.44
H-Ac-Ep	24.68

The minimum-cost path required a total delta-v of 9.47 km/s; the highest-cost path through three of the four asteroids had a delta-v of 25.13 km/s.

Figure 9 shows the minimum delta-v trajectory.

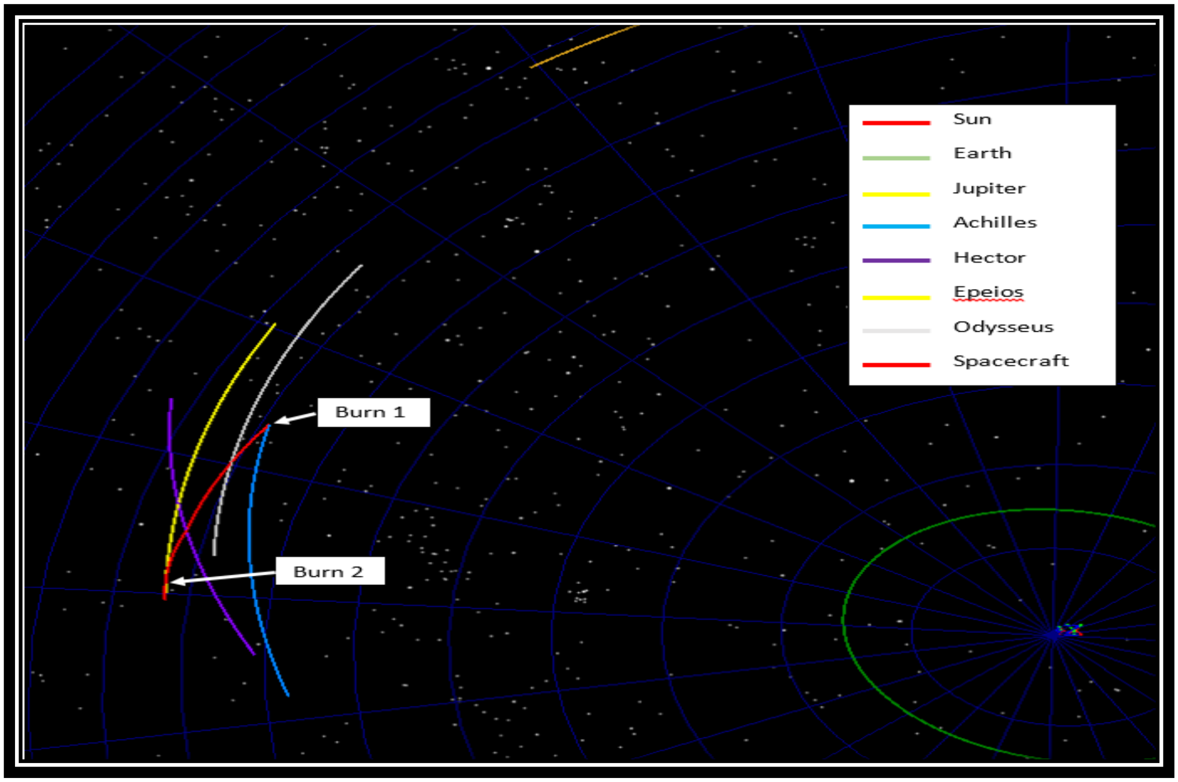


Figure 9. Minimum delta-v trajectory through three Trojan asteroids.

The two impulsive burns for the optimal Achilles-Hector-Epeios path are given in Table 9.

Table 9: Burn vectors for the optimal path through the three Trojan asteroids.

	X (km/s)	**Y (km/s)**	**Z (km/s)**	**Total |delta-v| (km/s)**
Achilles to Hektor	-0.6176	2.233	5.233	5.73
Hector to Epeios	-0.1873	3.658	-0.2592	3.67

Exhaustive Search: 8 Asteroids with Low Eccentricity and Inclination

Results

Cases provide an initial estimate of the approximate delta v required for a mission to multiple Trojan asteroids. The target asteroids were selected from the small number of named Trojan asteroids, which are among the largest and best characterized. However, these asteroids have widely varying orbital properties, such that spacecraft transfers between them are difficult. Even for the best trajectories, the calculated delta-v for a single asteroid-to-asteroid transfer is greater than 3 km/s. With current space propulsion systems, a realistic mission trajectory would require less than 3 km/s delta-v for the entire interplanetary trajectory (from Earth departure), and ideally less than 2 km/s delta-v for the intra-Trojan portion.

To improve upon the initial results, a different set of eight asteroids was selected with more closely-matched orbital parameters. The eight Trojan asteroids, 89913 (2002 EC24) (B), 228108 (2008 SU277) (M), 263795 (2008 QP 41) (S), 316146 (2009 SV347) (Si), Achilles (Ac), 624 Hektor (H), 2148 Epeios (Ep), and 1143 Odysseus (O) were selected for low orbit eccentricity and inclination.

Three problem cases were considered. In Case 1, the trajectory was required to include all eight asteroids in the set ($m = n$). In Case 2, the trajectory was required to include only four out of the eight asteroids ($m < n$) through optimal path. In Case 3, the trajectory was required to include only three out of the eight asteroids ($m < n$) through optimal path. Mission date for this transfer is starting from 1 January 2012 with transfer time increment of 200 days between each asteroid. Hence plotted trajectory transfer consists of mission dates as 1 January 2013, 20 July 2013, 4 February 2014, 23 August 2014, 11 March 2015, 27 September 2015, 14 April 2016, and 31 October 2016 respectively.

Case 1

The MATLAB-GMAT algorithm calculated the total delta-v for each of the 1680 possible paths through all eight asteroids. The five trajectories with the smallest delta-v are shown in Table 10.

The unnamed asteroids are designated as follow:

89913 (2002 EC24) – B

228108 (2008 SU277) – M

263795 (2008 QP 41) – S

316146 (2009 SV347) – Si

Table 10: Five minimal delta-v trajectories through eight Trojan asteroids: 89913 (2002EC24) (B), 228108 (2008 SU277) (M), 263795 (2008 QP 41) (S), 316146 (2009 SV347) (Si), Achilles (Ac), 624 Hektor (H), 2148 Epeios (Ep), and 1143 Odysseus (O).

Trojan Tour Order	**Total |delta-v| (km/s)**
S-H-M-Si-Ac-O-Ep-B	5.86
M-H-S-Si-ac-Ep-O-B	6.05
S-H-Si-M-Ep-O-Ac-B	6.13
H-S-Si-M-O-Ep-Ac-B	6.99
M-H-S-Ep-Si-Ac-O-B	7.3

The minimum-cost path required a total delta-v of 5.86 km/s; the highest-cost path through the same eight asteroids had a delta-v of 68.64 km/s.

The three impulsive burns for the optimal path from 263795 (2008 QP 41) (S)-Hektor-228108 (2008 SU277) (M) - 316146 (2009 SV347) (Si) are given in Table 11.

Table 11: Burn vectors for the optimal path through eight Trojan asteroids considering only three burns.

	X (km/s)	**Y (km/s)**	**Z (km/s)**	**Total |delta-v| (km/s)**
S to Hektor	-0.1445	-0.8129	-0.6201	1.033
Hektor to M	1.2171	-0.6192	-1.1755	1.805
M to Si	0.1956	2.5196	1.9693	3.20

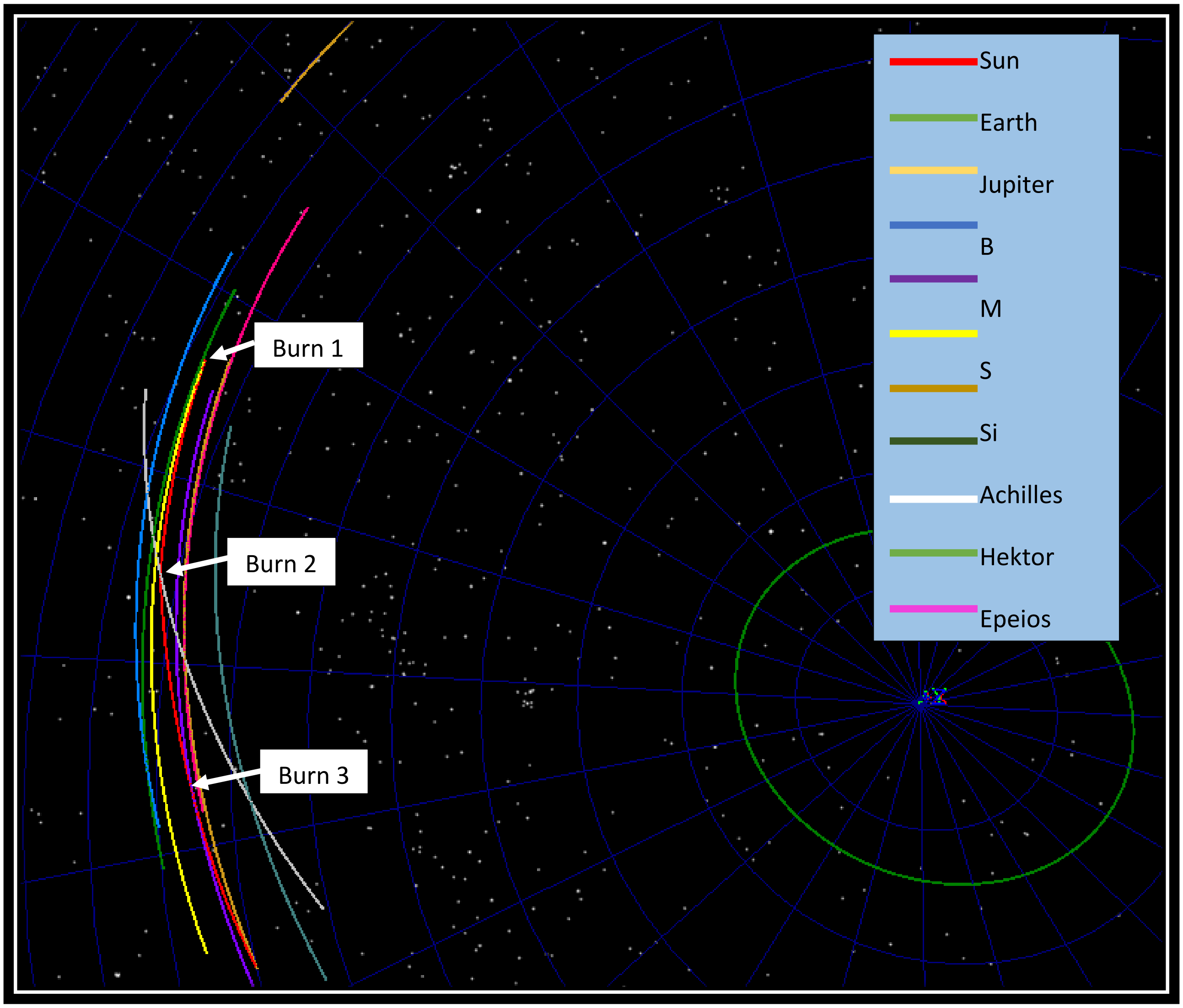


Figure 10. Minimal delta-v trajectory through eight Trojan asteroids.

Case 2

The MATLAB-GMAT algorithm calculated the total delta-v for a trajectory through all permutations of four out of the eight four asteroids. The trajectories with the smallest delta-v are shown in Table 12.

Table 12: Minimal delta-v trajectory through four out of the eight Trojan asteroids: S (263795 (2008 QP 41)), 624 Hektor (H), M (228108 (2008 SU277)), and Si (316146 (2009 SV347)).

Trojan Tour Order	Total |delta-v| (km/s)
S-H-M-Si	5.86

Figure11. Shows the minimum delta-v trajectory.

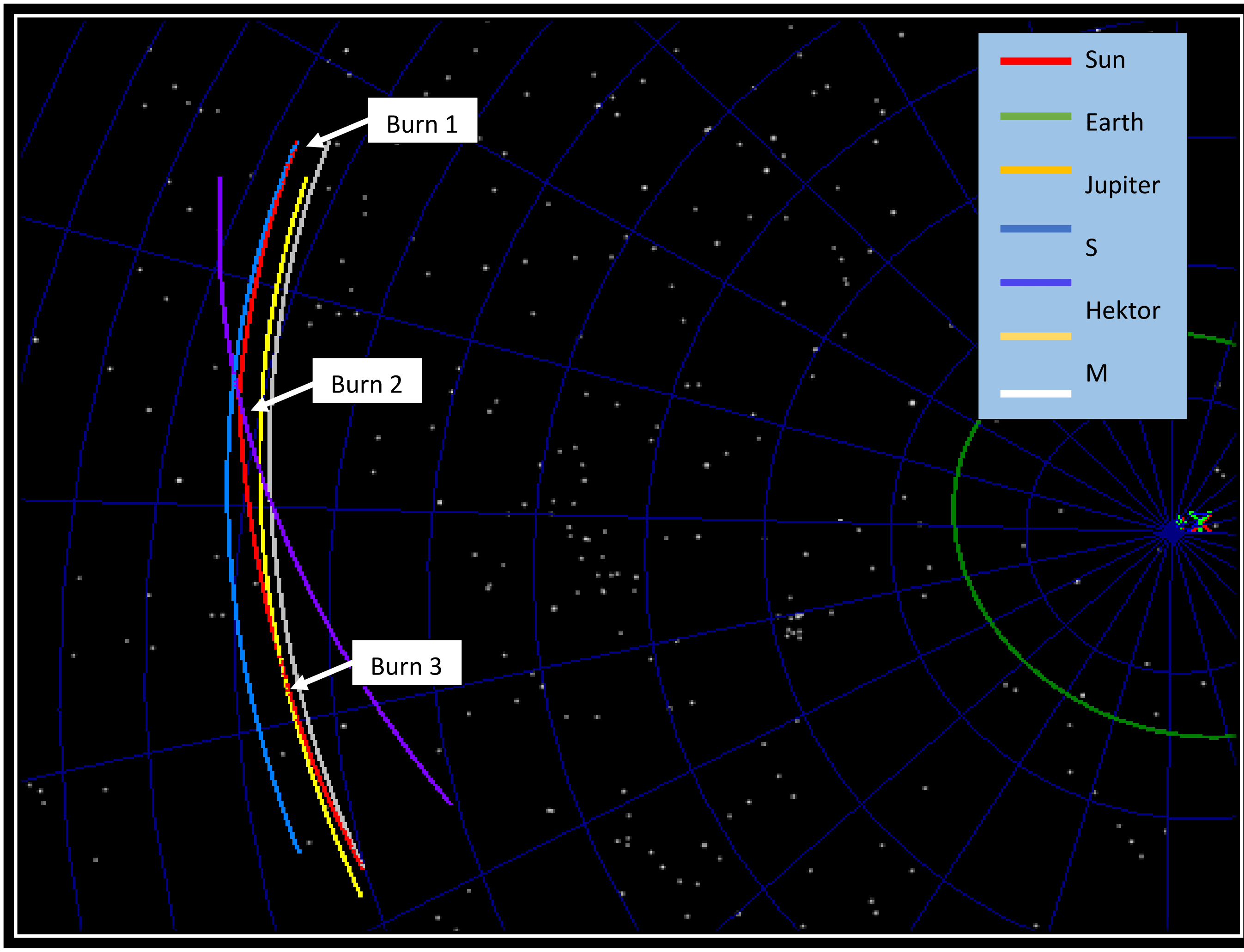


Figure 11. Minimal delta-v trajectory through four Trojan asteroids.

The minimum-cost path required a total delta-v of 5.86 km/s. The largest trajectory change occurs on the second burn, as the spacecraft maneuvers to encounter 2281 (2008 SU277). During the 2013 timespan considered in this analysis, the position of 2281 (2008 SU277) is distant from the remaining three asteroids in the set.

Case 3

The MATLAB-GMAT algorithm calculated the total delta-v for a trajectory through all permutations of three out of the eight asteroids. The five trajectories with the smallest delta-v are shown in Table 13. Of these optimal trajectories, none include asteroid (316146 (2009 SV347) (Si)).

Table 13: Five minimal delta-v trajectories through three out of the eight Trojan asteroids: S (263795 (2008 QP 41)), 624 Hektor (H), M (228108 (2008 SU277)), and Si (316146 (2009 SV347)).

Trojan Tour Order	**Total |delta-v| (km/s)**
M-H-S	2.153
S-H-M	2.779
H-S-M	5.899
S-M-H	7.622
M-S-H	8.927

The minimum-cost path required a total delta-v of 2.153 km/s; the highest-cost path through three of the four asteroids had a delta-v of 9.573 km/s.

Figure 12 shows the minimum delta-v trajectory.

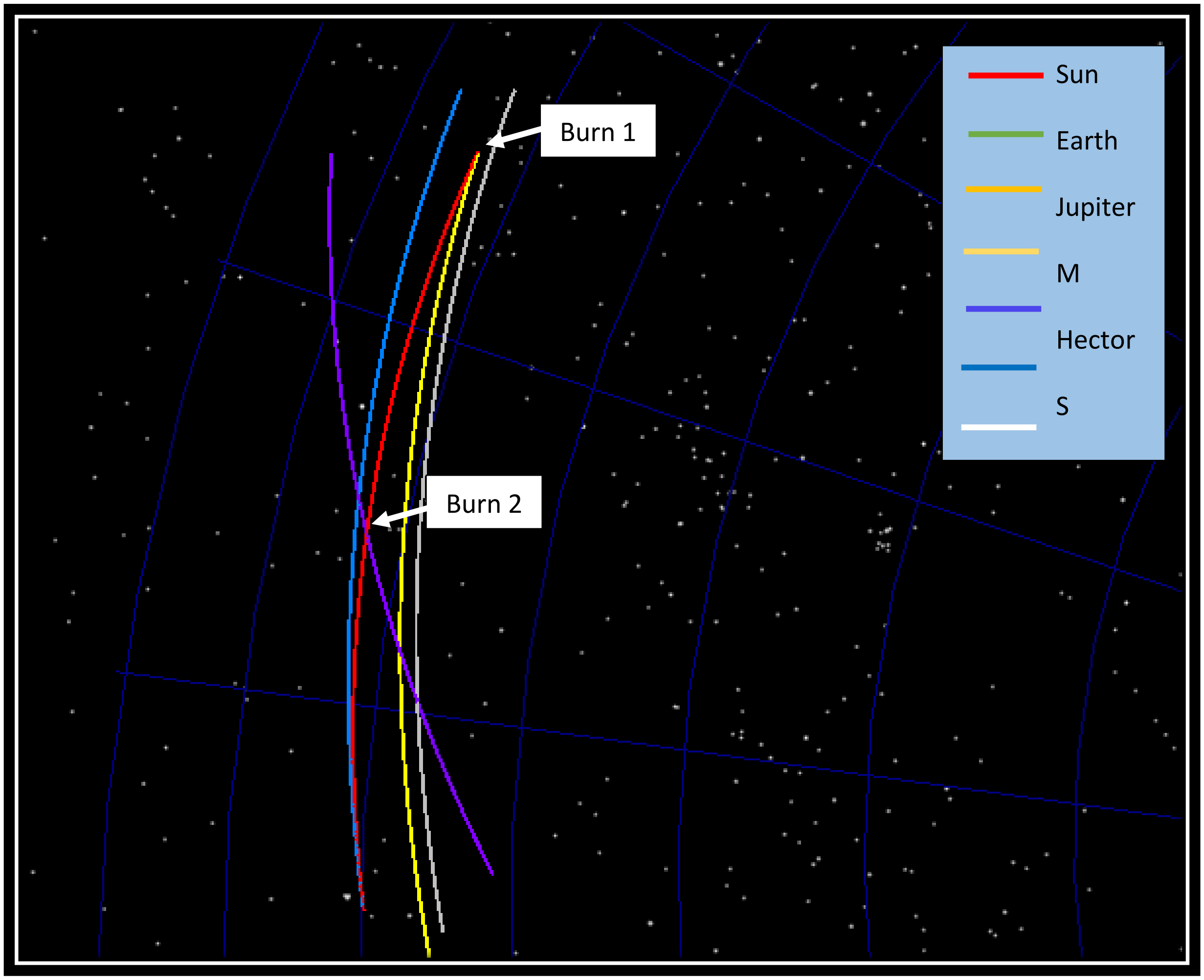


Figure 12. Minimum delta-v trajectory through three Trojan asteroids.

The results of two impulsive burns and optimal path for M (228108 (2008 SU277) (M)), 624 Hektor (H), and S (263795 (2008 QP 41) (S)) are as follows.

Table 14: Burn vectors for the optimal path through the three Trojan asteroids.

		X (km/s)	Y (km/s)	Z (km/s)	Total |delta-v| (km/s)
M to Hektor		-0.2902	1.0361	1.3551	1.58
Hector to S		-0.247	0.086	-0.27	0.15

Serial Rendezvous Search

RESULTS

The preceding analyses assumed a fixed time of departure from the first asteroid. However, the asteroid traveling salesman problem is highly time-dependent; the relative positions and velocities of the asteroids are time-varying. As such, any tour through a given set of asteroids may require different total delta-v if performed at different dates and/or over different time spans.

In this section, the problem was considered as trajectory transfer with varying transfer dates and times. In this case the eight low inclination/eccentricity asteroids from Section c were considered using the Serial Rendezvous Search method. The crucial deciding factor for varying transfer dates and times is the value of orbital inclination (i) and eccentricity (e). The asteroids which having less value of ‘i’ and ‘e’ are selected in order to achieve minimal delta-v for each trajectory transfer.

Case 1 assumes a trajectory starting from asteroid 263795 (2008 QP 41) – S. Case 2 assumes a start from asteroid 588 Achilles.

Case 1

The MATLAB-GMAT algorithm calculated the total delta-v for each path by fixing the starting asteroid. The minimal delta-v of these paths with varying transfer dates and times is utilized for next burn. The criterion for increasing start date is varied in steps of 60 and 120 days respectively. Whereas flight duration time was increased by 50, 100 and 150 days respectively.

Eight Trojan asteroids are as: 89913 (2002 EC24) (B), 228108 (2008 SU277) (M), 263795 (2008 QP 41) (S), 316146 (2009 SV347) (Si), Achilles (Ac), 624 Hektor (H), 2148 Epeios (Ep), and 1143 Odysseus (O).

Table 15 summarizes the best result found.

Table 15: Burn vectors for the optimal path through the Eight Trojan asteroids.

Burn Order	**Trojan Order**	**Total |delta-v| (km/s)**	**Starting Date**	**Increased start date**	**Flight Duration**	**Flight Time**
1	S - Hektor	1.42	1 Jan 2013	60 Days	2 Mar to 30 Jul 2013	150 Days
2	Hektor – Si	3.42	30 Jul 2013	-	30 Jul to 7 Nov 2013	100 Days
3	Si – M	2.14	7 Nov 2013	120 Days	7 Mar to 4 Aug 2014	150 Days
4	M – Achilles	10.78	4 Aug 2014	-	4 Aug 2014 to 1 Jan 2015	150 Days
5	Achilles - Odysseus	20.48	1 Jan 2015	120 Days	1 May to 28 Sep 2015	150 Days
6	Odysseus - B	4.1	28 Sep 2015	-	28 Sep 2015 to 25 Feb 2016	150 Days
Last Burn B to Epeios **Optimal Path is S-Hektor-Si-M-Achilles-Odysseus-B-Epeios**						

Four of the asteroid-to-asteroid transfer results obtained is within 5 Km/s.

Table 16 : Results for all transfer dates (start at S).

* First Burn *			
Asteroid Transfer	**Transfer Dates and Times**		
	1 Jan 2013 to 20 Feb 2013 (50 days)	**1 Jan 2013 to 11 Apr 2013 (100 days)**	**1 Jan 2013 to 31 May 2013 (150 days)**
S-Hektor	14.23	4.88	1.91
S-M	8.91	4.77	3.39
S-Si	6.427	3.38	2.39
S-Achilles	19.86	10.95	7.52
S-Odysseus	44.78	22.15	14.6
S-Epeios	22.24	11.09	7.41
S-B	28.05	14.39	9.84
Changing time span	**Increasing Start date by 60 Days**		
	2 Mar 2013 to 21 Apr 2013 (50 days)	**2 Mar 2013 to 10 Jun 2013 (100 days)**	**2 Mar 2013 to 30 Jul 2013 (150 days)**
S-Hektor	8.87	2.52	1.42
S-M	9.64	5.141	3.65
S-Si	6.88	3.65	2.59
S-Achilles	22.3	12.25	8.93
S-Odysseus	44.01	27.71	14.29
S-Epeios	22.15	11.08	7.42
S-B	28.84	14.76	10.09
Changing time span	**Increasing Start date by 120 Days**		
	1 May 2013 to 20 Jun 2013(50 days)	**1 May 2013 to 9 Aug 2013 (100 days)**	**1 May 2013 to 28 Sep 2013(150 days)**
S-Hektor	4.48	2.35	2.8
S-M	10.39	5.52	3.9
S-Si	7.42	3.96	2.82
S-Achilles	24.88	13.57	9.81
S-Odysseus	43.13	21.24	13.95

Table 16 - continued

S-Epeios	22.12	11.11	7.47
S-B	29.6	15.13	10.31
# Second Burn #			
Asteroid Transfer	**Transfer Dates and Times**		
	30 Jul 2013 to 18 Sep 2013 (50 days)	**30 Jul 2013 to 7 Nov 2013 (100 days)**	**30 Jul 2013 to 27 Dec 2013 (150 days)**
Hektor-Si	4.91	3.42	3.5
Hektor-M	6.914	4.71	4.32
Hektor-Achilles	21.71	10.49	6.79
Hektor-Odysseus	43.44	21.08	13.87
Hektor-Epeios	24.64	12.95	9.53
Hektor-B	34.39	17.93	12.68
Changing time span	**Increasing Start date by 60 Days**		
	28 Sep 2013 to 17 Nov 2013(50 days)	**28 Sep 2013 to 6 Jan 2014 (100 days)**	**28 Sep 2013 to 25 Feb 2014(150 days)**
Hektor-Si	7.49	5.62	5.11
Hektor-M	10.03	6.82	5.85
Hektor-Achilles	20.77	10.07	6.57
Hektor-Odysseus	41.83	20.68	13.88
Hektor-Epeios	26.27	14.59	11.04
Hektor-B	36.13	19.19	13.74
Changing time span	**Increasing Start date by 120 Days**		
	27 Nov 2013 to 16 Jan 2014(50 days)	**27 Nov 2013 to 7 Mar 2014(100 days)**	**27 Nov 2013 to 26 Apr 2014(150 days)**
Hektor-Si	12	8.05	6.76
Hektor-M	14.35	9.13	7.42
Hektor-Achilles	19.98	9.77	6.42
Hektor-Odysseus	41.2	20.78	14.23
Hektor-Epeios	29.83	16.94	12.86

Table 16 - continued

Hektor-B	38.8	20.84	15
* Third Burn *			
Asteroid Transfer	**Transfer Dates and Times**		
	7 Nov 2013 to 27 Dec 2013 (50 days)	**7 Nov 2013 to 15 Feb 2014 (100 days)**	**7 Nov 2013 to 6 Apr 2014(150 days)**
Si-M	7.64	3.69	2.37
Si-Achilles	25.79	13.59	9.52
Si-Odysseus	41.39	20.69	13.81
Si-Epeios	27.63	14.53	10.2
Si-B	36.47	18.95	13.11
Changing time span	**Increasing Start date by 60 Days**		
	6 Jan 2014 to 25 Feb 2014 (50 days)	**6 Jan 2014 to 16 Apr 2014 (100 days)**	**6 Jan 2014 to 5 Jun 2014(150 days)**
Si-M	7.31	3.51	2.25
Si-Achilles	27.36	14.34	9.99
Si-Odysseus	41.23	20.61	13.77
Si-Epeios	29.31	15.4	10.79
Si-B	38.09	19.75	13.65
Changing time span	**Increasing Start date by 120 Days**		
	7 Mar 2014 to 26 Apr 2014(50 days)	**7 Mar 2014 to 15 Jun 2014(100 days)**	**7 Mar 2014 to 4 Aug 2014 (150 days)**
Si-M	6.97	3.35	2.14
Si-Achilles	28.84	15.04	10.43
Si-Odysseus	41.09	20.55	13.73
Si-Epeios	31.05	16.29	11.38
Si-B	39.68	20.53	14.15

Table 16 - continued

# Fourth Burn #			
Asteroid Transfer	Transfer Dates and Times		
	4 Aug 2014 to 23 Sep 2014 (50 days)	4 Aug 2014 to 12 Nov 2014 (100 days)	4 Aug 2014 to 1 Jan 2015(150 days)
M-Achilles	29.99	15.6	10.78
M-Odysseus	46.18	22.92	15.2
M-Epeios	39.7	20.41	13.98
M-B	48	24.5	16.61
Changing time span	3 Oct 2014 to 22 Nov 2014 (50 days)	3 Oct 2014 to 11 Jan 2015 (100 days)	3 Oct 2014 to 2 Mar 2015(150 days)
M-Achilles	31.35	16.2	11.13
M-Odysseus	45.63	22.65	15.01
M-Epeios	40.92	20.99	14.34
M-B	48.96	24.91	16.9
Changing time span	Increasing Start date by 120 Days		
	2 Dec 2014 to 21 Jan 2015(50 days)	2 Dec 2014 to 12 Mar 2015(100 days)	2 Dec 2014 to 1 May 2015 (150 days)
M-Achilles	32.53	16.72	11.42
M-Odysseus	45.1	22.39	14.85
M-Epeios	42.06	21.52	14.64
M-B	49.85	25.33	17.16
# Fifth Burn #			
Asteroid Transfer	Transfer Dates and Times		
	1 Jan 2015 to 20 Feb 2015 (50 days)	1 Jan 2015 to 11 Apr 2015(100 days)	1 Jan 2015 to 31 May 2015(150 days)
Achilles-Odysseus	61.91	31	20.7
Achilles-Epeios	67.68	34.71	23.69

Table 16 - continued

Changing time span	Increasing Start date by 60 Days		
	2 Mar 2015 to 21 Apr 2015 (50 days)	2 Mar 2015 to 10 Jun 2015(100 days)	2 Mar 2015 to 30 Jul 2015(150 days)
Achilles-Odysseus	61.82	30.91	20.61
Achilles-Epeios	69.56	35.53	24.15
Achilles-B	70.07	35.34	23.74
Changing time span	**Increasing Start date by 120 Days**		
	1 May 2015 to 20 Jun 2015(50 days)	1 May 2015 to 9 Aug 2015(100 days)	1 May 2015 to 28 Sep 2015 (150 days)
Achilles-Odysseus	61.63	30.77	20.48
Achilles-Epeios	71.14	36.19	24.51
Achilles-B	70.6	35.51	23.79
# Sixth Burn #			
Asteroid Transfer	**Transfer Dates and Times**		
	28 Sep 2015 to 17 Nov 2015 (50 days)	28 Sep 2015 to 6 Jan 2016(100 days)	28 Sep 2015 to 25 Feb 2016(150 days)
Odysseus-Epeios	21.74	11.23	7.73
Odysseus-B	11.89	6.05	4.1
Changing time span	**Increasing Start date by 60 Days**		
	27 Nov 2015 to 16 Jan 2016 (50 days)	27 Nov 2015 to6 Mar 2016 (100 days)	27 Nov 2015 to 25 Apr 2016 (150 days)
Odysseus-Epeios	22.54	11.61	7.96
Odysseus-B	12.14	6.18	4.19
Changing time span	**Increasing Start date by 120 Days**		
	26 Jan 2016 to 16 Mar 2016(50 days)	26 Jan 2016 to 5 May 2016 (100 days)	26 Jan 2016 to 24 Jun 2016 (150 days)

Table 16 - continued

Odysseus-Epeios	23.28	11.96	8.18
Odysseus-B	12.38	6.3	4.27
Last Burn - B to Epeios			
Optimal Path is S-Hektor-Si-M-Achilles-Odysseus-B-Epeios			

Case 2

Case 2 uses the same approach as Case 1 with a different start asteroid: 588 Achilles.

The MATLAB-GMAT algorithm calculated the total delta-v for each path by fixing the starting asteroid. The minimal delta-v of these paths with varying flight time duration is utilized for next burn. The flight duration time is increased in steps of 50, 100, 150 and 200 days respectively.

Eight Trojan asteroids are as: 89913 (2002 EC24) (B), 228108 (2008 SU277) (M), 263795 (2008 QP 41) (S), 316146 (2009 SV347) (Si), Achilles (Ac), 624 Hektor (H), 2148 Epeios (Ep), and 1143 Odysseus (O).

Table 16 shows all the date and time ranges evaluated.

Table 17: Burn vectors for the optimal path through the Eight Trojan asteroids.

Burn Order	Trojan Order	Total |delta-v| (km/s)	Starting Date	Flight Duration	Flight Days
1	Achilles – M	7.063	1 Jan 2013	15 Feb 2013 to 26 May 2013	100 Days
2	M – Si	1.95	26 May 2013	26 May 2013 to 12 Dec 2013	200 Days
3	Si – S	5.42	12 Dec 2013	12 Dec 2013 to 22 Mar 2014	100 Days
4	S – Epeios	6.24	22 Mar 2014	22 Mar 2014 to 8 Oct 2014	200 Days
5	Epeios – B	4.17	8 Oct 2014	8 Oct 2014 to 26 Apr 2015	200 Days
6	B - Odysseus	2.97	26 Apr 2015	26 Apr 2015 to 12 Nov 2015	200 days

Here almost all of the results obtained for trajectory transfer are within 5 Km/s.

Table 18 : Results for all transfer dates (start at Achilles)

Burn	Pair delta-v for 50 Days		Pair delta-v for 100 Days		Pair delta-v for 150 Days		Pair delta-v for 200 Days	
First Burn	1 Jan 2013 to 20 Feb 2013		15 Feb 2013 to 26 May 2013		1 Mar 2013 to 31 May 2013		1 Apr 2013 to 20 July 2013	
	Achilles-M	11.1	Achilles-M	7.063	Achilles-M	9.59	Achilles-M	9.55
	Achilles-S	19.86	Achilles-S	11.92	Achilles-S	12.96	Achilles-S	12.23
	Achilles-Hektor	25.79	Achilles-Hektor	11.94	Achilles-Hektor	12.37	Achilles-Hektor	11.25
	Achilles-Si	17.95	Achilles-Si	10.06	Achilles-Si	12.04	Achilles-Si	11.35
	Achilles-Odysseus	58.32	Achilles-Odysseus	29.6	Achilles-Odysseus	25.28	Achilles-Odysseus	21.25
	Achilles-Epeios	35.7	Achilles-Epeios	19.3	Achilles-Epeios	18.25	Achilles-Epeios	16.39
	Achilles-B	45.43	Achilles-B	24.75	Achilles-B	22.03	Achilles-B	19.22
Second Burn	26 May 2013 to 15 Jul 2013		26 May 2013 to 3 Sep 2013		26 May 2013 to 23 Oct 2013		26 May 2013 to 12 Dec 2013	
	M-S	10.7	M-S	5.68	M-S	4.01	M-S	3.17
	M-Hektor	7.19	M-Hektor	3.29	M-Hektor	2.85	M-Hektor	2.98
	M-Si	8.51	M-Si	4.14	M-Si	2.68	M-Si	1.95
	M-Odysseus	50.15	M-Odysseus	24.95	M-Odysseus	16.57	M-Odysseus	12.39
	M-Epeios	30.14	M-Epeios	15.6	M-Epeios	10.78	M-Epeios	8.41
	M-B	39.94	M-B	20.28	M-B	13.94	M-B	10.77
Third Burn	12 Dec 2013 to 31 Jan 2014		12 Dec 2013 to 22 Mar 2014		12 Dec 2013 to 11 May 2014		12 Dec 2013 to 30 June 2014	
	Si-S	10.16	Si-S	5.42	Si-S	10.44	Si-S	7.28
	Si-Hektor	13.2	Si-Hektor	8.66	Si-Hektor	11.9	Si-Hektor	9.13
	Si-Odysseus	41.29	Si-Odysseus	20.64	Si-Odysseus	27.19	Si-Odysseus	18.14
	Si-Epeios	28.6	Si-Epeios	15.04	Si-Epeios	21.39	Si-Epeios	14.76
	Si-B	37.42	Si-B	19.42	Si-B	25.87	Si-B	17.72
Fourth Burn	22 Mar 2014 to 11 May 2014		22 Mar 2014 to 30 June 2014		22 Mar 2014 to 9 Aug 2014		22 Mar 2014 to 8 Oct 2014	
	S-Hektor	28.71	S-Hektor	16.58	S-Hektor	12.53	S-Hektor	10.5
	S-Odysseus	37.2	S-Odysseus	18.12	S-Odysseus	11.79	S-Odysseus	8.6
	S-Epeios	23.44	S-Epeios	11.94	S-Epeios	8.13	S-Epeios	6.24
	S-B	32.75	S-B	16.58	S-B	11.2	S-B	8.5
Fifth Burn	8 oct 2014 to 27 Nov 2015		8 oct 2014 to 16 Jan 2015		8 Oct 2014 to 7 Mar 2015		8 oct 2014 to 26 Apr 2015	
	Epeios-Hektor	59.6	Epeios-Hektor	32.54	Epeios-Hektor	23.37	Epeios-Hektor	18.17
	Epeios-Odysseus	17.47	Epeios-Odysseus	8.92	Epeios-Odysseus	6.14	Epeios-Odysseus	4.77
	Epeios-B	12.58	Epeios-B	6.928	Epeios-B	5.08	Epeios-B	4.17
Sixth Burn	26 Apr 2015 to 15 Jun 2015		26 Apr 2015 to 4 Aug 2015		26 Apr 2015 to 23 Sep 2015		26 Apr 2015 to 12 Nov 2015	
	B-Hektor	75.1	B-Hektor	39.3	B-Hektor	27.34	B-Hektor	21.34
	B-Odysseus	11.4	B-Odysseus	5.76	B-Odysseus	3.89	B-Odysseus	2.97
Last Burn - Odysseus to Hektor								
Optimal Path is Achilles-M-Si-S-Epeios-B-Odysseus-Hektor								

V. CONCLUSION AND FUTURE WORK

This thesis presents the development of preliminary methods and software algorithms to search for an optimum trajectory for a mission to the Jovian Trojan asteroids. Solutions were first obtained for optimized trajectory transfers between four Trojan asteroids, Achilles, Hektor, Agamemnon, and Nestor, near the Jupiter-Sun libration point L4. Furthermore by introducing eight asteroids, the results for optimized trajectory transfers were evaluated. The approach of varying transfer dates and time is also scrutinized for optimized trajectory transfers between eight asteroids. For accurate trajectory optimization and visualization of the trajectory transfer, these results are explored in GMAT software.

Trajectory optimization for a Trojan asteroid tour and rendezvous mission is a complex problem with many locally-optimum solutions. Due to the large flight distance and long time span required to reach the Trojans, selection of a minimum-fuel trajectory may be a critical factor in determining feasibility and scientific value of a Trojan mission. A dynamic programming approach enables systematic evaluation of the solution space. The traveling salesman algorithm, combined with the GMAT tool for evaluating the cost of individual trajectories, provides a framework to determine an optimal solution to this highly complex problem. The results achieved enable us to visualize the motion of planets, asteroids, and spacecraft along with the optimized trajectory transfer.

When a larger set of asteroids is considered, the number of permutations becomes prohibitively large for the exhaustive search method. The serial rendezvous search method is more efficient for larger search sets, and also enables variation of transfer dates and times.

Future work on this problem may use successive approximation schemes from dynamic programming. Sub problems that contain transfers between the Se pairs of asteroids could be solved only once, reducing computational time. Pruning assumptions, such as limiting the search to Trojan asteroids with low orbital inclinations, can also improve the results for large search sets.

I recommend the continued study of mission design for the Trojan asteroids tour, as early results indicate that a well-instrumented mission is feasible. The Trojan asteroids are unmapped and fundamentally unknown in many aspects. A spacecraft mission to the Trojan asteroids will help in understanding various enigmatic features of these bodies and evolution of the solar system as whole. The spacecraft flybys give us general information about composition, geology and density while a mission that orbits a Trojan asteroid will provide us with facts about interior and exterior properties. The trajectory of this mission will strongly influence the value of result acquired.

Rivkin et al. [1] in their study demonstrated that "The technical feasibility for any of these missions is well within our capabilities at the present time. Past missions to asteroids and recent developments in low-cost and long-duration cruise operations are two successes that NASA can build on to realize a Trojan asteroid mission in the next decade. At a minimum, we strongly support the continued inclusion of a Trojan-focused mission in the New Frontiers list of eligible missions". A Trojan tour mission will surely improve the space research community's understanding of solar system.

REFERENCES

[1]. Brent W. Barbee, George W. Davis, and Sun Hur–Diaz “Spacecraft trajectory design for tours of multiple small bodies.” AAS 09-433.

[2]. S. Diniega, , K. Sayanagi, J.Balcerski, B. Carande, R. Diaz-Silva , A. Fraeman, S. Guzewich, J. Hudson, A. Nahm, S. Potter-McIntyre, M. Route, K. Urban S. Vasisht, B. Benneke, S. Gil, R. Livi, , B. Williams, C. Budney, L.

[3]. “Design and Optimization of Low-Thrust Trajectories with Gravity Assists” T. Troy McCo-naghy, Theresa J. Debban, Anastassios E. Petropoulos, James M. Longuski Journal of Spacecraft and Rockets, 2003, Vol.40: 380-387, 10.2514/2.3973

[4]. “Trajectories to Jupiter via Gravity Assists from Venus, Earth, and Mars” Anastassios E. Petropoulos, James M. Longuski, Eugene P. Bonfiglio Journal of Spacecraft and Rockets, 2000, Vol.37: 776-783, 10.2514/2.3650

[5]. D. Izzo, V. M. Becerra, D. R. Myatt, S. J. Nasuto, and J. M. Bishop. "Search space pruning and global optimization of multiple gravity assist spacecraft trajectories." Journal of Global Optimization 38, no. 2 (2007): 283-296.

[6].http://gmat.gsfc.nasa.gov

[7]. T. Grav, A.K. Mainzer, J.M. Bauer, J.R. Masiero, and C.R. Nugent. “WISE/NEOWISE Ob-servations of the Jovian Trojan Population: Taxonomy.” The Astrophysical Journal, Vol. 759, No. 1, 2012.

[8]. T. Grav, , A. K. Mainzer, J. Bauer, J. Masiero, T. Spahr, R. S. McMillan, R. Walker et al. "WISE/NEOWISE Obser-vations of the Jovian Trojans: Preliminary Results." The Astro-physical Journal, Vol 742, No. 1, 2011.

[9]. Marcus Langston, Matt Sorgenfrei, Ricardo Diaz-Silva, Federico Aguilar, and Alvaro Ibaseta Fidalgo "A Mission Architecture for Scientific Exploration of a Trojan Object using Solar Electric Propulsion." AIAA SPACE

[10]. http://ssd.jpl.nasa.gov/sbdb_query.cgi

[11]. Andrew S. Rivkin (JHU/APL), Joshua Emery (U. Tennessee), Antonella Barucci (Observatoire de Paris), James F. Bell (Cornell University), William F. Bottke (SwRI), Elisabetta Dotto (Osservatorio Astronomico di Roma), Robert Gold (JHU/APL), Carey Lisse (JHU/APL), Javier Licandro (Instituto de Astrofísica de Canarias), Louise Prockter (JHU/APL), Charles Hibbits (JHU/APL), Michael Paul (Applied Research Laboratory, Penn State University), Alessondra Springmann (MIT), Bin Yang (University of Hawaii) "The Trojan Asteroids: Keys to Many Locks"

[12].Bernardetta Addis, Andrea Cassioli, Marco Locatelli, and Fabio Schoen "A global optimization method for the design of space trajectories." springer science, LLC 2009, Comput. Optim. Appl. 48, 635–652 (2011).

[13]. M. A. Barucci, D. P. Cruikshank and S. Mottola "Physical Properties of Trojan and Centaur Asteroids." The Astronomical Journal, Volume 141, Issue 5, article id. 170, 32 pp. (2011).

[14]. Massimiliano Vasile and Marco Locatelli "A hybrid multiagent approach for global trajectory optimization." July 9, 2008. Springer Science, J Glob. Optim. 44, 461–479 (2009).

[15]. Howard D. Curtis "orbital mechanics for Engineering Students" Second Edition, Elsevier publication.

[16]. Giovanni Stracquadanio, Angelo La Ferla, Matteo De Felice and Giuseppe Nicosia "Design of Robust Space Trajectories." M. Bramer et al. (eds.), Research and Development in Intelligent Systems XXVIII, Springer-Verlag London Limited 2011

[17]. Jon A. Sims, Paul A. Finlayson, Edward A. Rinderle, Matthew A. Vavrina and Theresa D. Kowalkowski "Implementation of a Low-Thrust Trajectory Optimization Algorithm for Preliminary Design." American Institute of Aeronautics and Astronautics, AIAA 2006-6746.

[18]. Book by Richard Bellman "Dynamic Programming" 1957

[19]. Camilla Colombo, Massimiliano Vasile and Gianmarco Radice "Optimal low-thrust trajectories to asteroids through an algorithm based on differential dynamic programming." Springer Science B.V. 2009, Celest. Mech. Dyn. Astr. 105, 75–112 (2009).

[20]. R. Bellman. "Dynamic Programming Treatment of the Travelling Salesman Problem." Journal of the ACM (JACM), Vol. 9, Issue 1, pp. 61-63, 1962.

http://www.credoreference.com/entry.do?id=6001394

[21]. http://naif.jpl.nasa.gov/naif/specificspicekernel.html accessed on 10/11/2013

[22]. http://gmat.gsfc.nasa.gov/project/overview.html accessed on 04/10/2013

APPENDIX

Reference link - http://ssd.jpl.nasa.gov/sbdb_query.cgi

Table 19 : Orbital configuration data for all asteroids under consideration.

No.	Designation (name)	Prov	Des.	Ln	M	Peri. (w)	Node (om)	Incl. (i)	e	a
1	(588) Achilles	1906	TG	L4	39.39	132.9	316.6	10.3	0.148	5.197
2	(624) Hektor	1907	XM	L4	337.37	179.9	342.8	18.2	0.024	5.245
3	(659) Nestor	1908	CS	L4	141.1	342.3	350.9	4.5	0.115	5.189
4	(911) Agamemnon	1919	FD	L4	72.54	81.3	338	21.8	0.068	5.267
5	(1143) Odysseus	1930	BH	L4	19.6	236.6	221.3	3.1	0.091	5.25
6	(1868) Thersites	2008	PL	L4	115.89	170.3	197.8	16.8	0.109	5.318
7	(2148) Epeios	1976	UW	L4	76.71	232.5	176.6	9.2	0.058	5.21
8	(2456) Palamedes	1966	BA1	L4	55.48	95.6	327.4	13.9	0.075	5.129
9	(2759) Idomeneus	1980	GC	L4	311.76	7.3	171.2	22	0.066	5.179
10	(3063) Makhaon	1983	PV	L4	3.59	203.7	287.9	12.2	0.06	5.198
11	89913	2002	EC24	L4	59.09	244.7	175.9	1.6	0.084	5.246
12	228108	2008	SU277	L4	42.55	210	238.8	1.6	0.092	5.19
13	263795	2008	QP41	L4	50.94	268	168.1	1.6	0.102	5.222
14	316146	2009	SV347	L4	40.19	208	241.1	1.6	0.083	5.102

Description of Matlab Code

The MATLAB code is developed to calculate the total delta-V for visiting a series of asteroids. The file *Trojan_transfer*, included below, is the primary driver file for the exhaustive search method. This example shows the case of finding the optimal trajectory through the four Trojan asteroids Achilles, Hektor, Nestor, and Agamemnon.

First, on lines 11-12 the list of asteroids to be visited is given in the cell array "targets". The NAIF identifications associated with each asteroid are given in a corresponding cell array in line 12.

Lines 14-17 specify the fixed time increment between asteroid flybys and the dates at which those flybys will occur. The dates are specified in two different formats for use by different functions. We assume the time increment between each asteroid transfer as 200 days. For the purpose of simply demonstrating the search method, a starting date of 1 January 2013 is selected.

Line 19 specifies the maximum number of iteration for GMAT targeting.

Lines 22-23 modify the MATLAB path to make use of the NAIF Mice Toolkit[1]. This is the MATLAB version of SPICE, a NASA software package and information system that includes space geometry and event data.

Lines 24-28 use Mice commands to load SPICE kernel files for the four target asteroids.

On lines 30-34, the matrix tour_order stores all possible permutations of the integers 1 to 4. The code will step through each line of this matrix to calculate every possible path.

[1] http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/MATLAB/

Tour_length will store the total delta-v for each path. The impulsive burn solution for each segment will be stored in dv_solns.

Lines 37-96 execute a 'for' loop to calculate the required delta-v for each segment of each path. The positions of the start and target asteroids at the appropriate dates are obtained from their spice kernels. The trajectory is initiated at the first asteroid, with spacecraft velocity equal to the asteroid's velocity.

The function *lambert_asteroids,* based on MATLAB code provided in [15], solves Lambert's problem between the pair of asteroids. The two function outputs are the initial and final velocities of the trajectory arc. The difference between the required initial velocity and the spacecraft's starting velocity is the delta-v that must be provided by the spacecraft engine.

In lines 58-65, the *lambert_asteroids* function is called twice, in the prograde and retrograde directions, to find the lower delta-v solution. The MATLAB solution to Lambert's problem is used to initiate the higher-fidelity GMAT optimizer.

Lines 79-83 set inputs and execute the function *Trojan_distance_GMAT*. This function performs up to 25 iterations of GMAT's trajectory optimization solver to find a precise delta-v solution for the transfer between the two asteroids. The solutions for each pair of asteroids are stored, and the total delta-v for each path through four asteroids (each row of tour_order) is calculated.

Lines 99-100 select the minimum delta-v path from all calculated permutations.

Lines 102-109 call another GMAT script, *Plot_Full_Mission*, which simulates the full trajectory between all four asteroids and outputs the final spacecraft position and velocity.

Trojan_transfer

```
1   % Calculate total delta-V for visiting a series of asteroids
2
3   % the list of asteroids to be visited is given in the cell array "targets"
4   % this program calculates all possible permutations of visiting order and
5   % the delta-V associated with each path.
6   % we assume the spacecraft starts each path at the first asteroid with
7   % velocity equal to the asteroid's velocity.
8
9   global maxiter
10
11  targets={'Achilles'; 'Hektor'; 'Nestor'; 'Agamemnon'};   % asteroid names
12  target_NAIF=[2000588; 2000624; 2000659; 2000911];        % NAIF id's associated with targets
13
```

```
time_inc = 200;                              % time increment between dates (days)
time_inc_s = time_inc*24*60^2;               % time increment between dates (seconds)
dates = {'Jan 1 2013'; 'July 20 2013'; 'Feb 4 2014'; 'Aug 23 2014'};
dates2 = {'01 Jan 2013'; '20 Jul 2013'; '04 Feb 2014'; '23 Aug 2014'};

maxiter = 25;                    % max number of GMAT targeting iterations

% Get spice kernels
addpath('C:\Users\User\Desktop\GMAT files\mice\src\mice\')
addpath('C:\Users\User\Desktop\GMAT files\mice\lib\')
cspice_furnsh( 'C:\Users\User\Desktop\GMAT files\Achilles.bsp' )
cspice_furnsh( 'C:\Users\User\Desktop\GMAT files\Hektor.bsp’)
cspice_furnsh( 'C:\Users\User\Desktop\GMAT files\Nestor.bsp’)
cspice_furnsh( 'C:\Users\User\Desktop\GMAT files\Agamemnon.bsp’)
cspice_furnsh( 'C:\Users\User\Desktop\GMAT files\naif0010.tls' )

inds=1:1:4;
```

```
tour_order=perms(inds);            % all possible permutations of 1:n
[p,q]=size(tour_order);            % p is the number of possible paths, q is the number of asteroids in each path
tour_length=zeros(p,1);            % tour_length will store the total delta-V for each path
dV_solns=zeros(3,q-1,p);           % dV_solns will store the impulsive burn solutions for each segment

% step through each row of the permutations list
for row=1:p
    d=0;
 % step through each segment of the path
    for seg=1:q-1;
    A = tour_order(row,seg);           % integer representing the start point
    etA = cspice_str2et(dates(seg));        % segment start time
     starg = mice_spkezr(int2str(target_NAIF(A)), etA, 'J2000', 'NONE', 'SUN'); % starting point
      state = [starg.state];
```

```
R1 = state(1:3);
v_ast = state(4:6);              % initial velocity of first asteroid
  if seg == 1
      v0 = v_ast;                 % s/c start at same velocity as first asteroid
      r0 = R1;
  end

  B = tour_order(row,seg+1);            % integer representing the target point
  etB = cspice_str2et(dates(seg+1));      % segment stop time
  ptarg = mice_spkpos(int2str(target_NAIF(B)), etB, 'J2000', 'NONE', 'SUN' );
  R2 = [ptarg.pos];                     % position of Asteroid 2


  % Solve Lambert problem in prograde direction
  string = 'pro';
  [V1a, V2a] = lambert_asteroids(R1, R2, time_inc_s, string);
    burna = V1a - v0;                        % Guess for burn: Lambert V1 - s/c initial
                                               velocity
```

```
% Solve Lambert problem in the retrograde direction
string = 'retro';
[V1b, V2b] = lambert_asteroids(R1, R2, time_inc_s, string);
 burnb = V1b - v0;                    % Guess for burn: Lambert V1 - s/c initial velocity

% Choose smaller burn
if norm(burnb)>norm(burna)
    burn = burna;
    V1 = V1a;
    V2 = V2a;
else
    burn = burnb;
    V1 = V1b;
    V2 = V2b;
end
```

```
% Call GMAT script to calculate actual burn
start_body = char(targets(A));
start_NAIF = target_NAIF(A);
targ_body = char(targets(B));
targ_NAIF = target_NAIF(B);
[burn_soln,r_end,v_end] = Trojan_distance_GMAT(start_body,targ_body,r0,v0,burn,start_NAIF,targ_NAIF,dates2(seg),time_inc);

d=d+norm(burn_soln);

% Store burn solution in 3-D array
dV_solns(:,seg,row)=burn_soln;

% Initial conditions for next segment
r0 = r_end;                    % s/c position at end of segment
```

```
        v0 = v_end;                    % s/c velocity at end of segment

    end
    tour_length(row)=d;
    end

% Look at minimum-delta-V path
[opt_path,opt_path_ind]=min(tour_length);   % length and index of optimal path
opt_tour_order=tour_order(opt_path_ind,:);  % optimal path (list of point numbers)

A = opt_tour_order(1);                % start point of opt path
etA = cspice_str2et(dates(1));    % segment start time
  starg = mice_spkezr(int2str(target_NAIF(A)), etA, 'J2000', 'NONE', 'SUN'); % starting point
state = [starg.state];
r0 = state(1:3);
v0 = state(4:6);                    % initial velocity of first asteroid
```

```
start_epoch=dates2(1);
[r_end,v_end] = Plot_Full_Mission(r0,v0,start_epoch,time_inc,dV_solns
(:,:,opt_path_ind));
```

Trojan_distance_GMAT

The following code is developed to calculate the actual burn with the help of GMAT script.

Lines 9-10 start the GMAT software and clears the previous data.

Lines 16-24 create the starting asteroid. GMAT is provided with the necessary spice kernel for information regarding space geometry and orientation data. The Sun is selected as the central body.

Lines 26-34 create the target asteroid. The target asteroid is also provided with space geometry and orientation data.

Lines 41-44 create the position of the Lagrange point L4, 60˚ ahead of Jupiter. The Sun is at the center and primary while Jupiter it the secondary body.

Lines 50-96 create the mission_spacecraft in the GMAT resource tab. Here the spacecraft is provided with a coordinate system along with various physical properties such as dry mass and coefficient of drag. The essential data is provided for the position of spacecraft along with its velocity component in X, Y and Z direction.

Lines 103-111 create the force model under the propagator tab in GMAT. The method used for error control is RSSStep and also solar radiation pressure is used.

Lines 117-125 select the defaultprop as propagator for the mission trajectory. The type of integrating method selected is RungeKutta89 with initial step size of 60 seconds. The mission will stop if the accuracy is violated.

Lines 131-138 create an impulsive burn as DefaultIB in the GMAT resource tab. This impulsive burn is required for the actual travel from the start asteroid to the target asteroid. The specific impulse is assumed constant and taken as 300 s. Gravitational acceleration is 9.81 m/s^2.

Lines 144-154 create the coordinate system with the Sun at the center. Similarly the co-ordinate system for orientation of the start body and the target body are created.

Lines 161-166 select the differential corrector as the solver using the forward difference method. It will continue working until the given number of iterations is reached.

Lines 172-201 create the orbital view for mission analysis. The Sun is selected as the point of reference. Different colors are assigned to the planets and asteroid bodies for easy visualization of the trajectory. This will give a clear picture of the trajectories of all the asteroids and Jupiter. Furthermore the path followed by the spacecraft to encounter the Trojan Asteroids is shown.

Lines 209-216 define the mission sequence. The X, Y, and Z components of the burn are allowed to vary within upper and lower bounds as the optimizer searches for the minimum delta-v trajectory.

Lines 224-230 return the result of the optimization. The MATLAB variable burn_soln contains the X, Y, and Z components of the delta-v vector.

Lines 231-243 give the final position and velocity associated with the spacecraft in the X, Y and Z directions. The variables burn_soln, r_end, and v_end are returned by the function.

Trojan_distance_GMAT

```
function [burn_soln,r_end,v_end] = Trojan_distance_GMAT(start_body,targ_body,r0,v0,burn,start_NAIF,targ_NAIF,start_epoch,time_inc)

% the inputs start_body and targ_body are strings
% the inputs r0, v0, and burn are 3x1 vectors of xyz velocities in km/s

global maxiter

%% Initialize GMAT
OpenGMAT();
 ClearGMAT();

 %----------------------------------------
 %---------- User-Defined Celestial Bodies
 %----------------------------------------
```

```
Create (['Asteroid ' start_body]);
GMAT ([start_body '.NAIFId = ' int2str(start_NAIF)]);
GMAT ([start_body '.OrbitSpiceKernelName = {"C:\Users\User\Desktop\GMAT files\' start_body '.bsp"}
GMAT ([start_body '.EquatorialRadius = 6378.1363']);
GMAT ([start_body '.Flattening = 0.0033527']);
GMAT ([start_body '.Mu = 398600.4415']);
GMAT ([start_body '.PosVelSource = "SPICE"']);
GMAT ([start_body '.CentralBody = "Sun"']);
GMAT ([start_body '.TextureMapFileName = "C:\Users\User\Desktop\GMAT files\GenericCelestialBody.jpg"']);

Create (['Asteroid ' targ_body]);
GMAT ([targ_body '.NAIFId = ' int2str(targ_NAIF)]);
GMAT ([targ_body '.OrbitSpiceKernelName = {"C:\Users\User\Desktop\GMAT files\' targ_body '.bsp"}']);
GMAT ([targ_body '.EquatorialRadius = 6378.1363']);
```

```
GMAT ([targ_body '.Flattening = 0.0033527']);
GMAT ([targ_body '.Mu = 398600.4415']);
GMAT ([targ_body '.PosVelSource = "SPICE"']);
GMAT ([targ_body '.CentralBody = "Sun"']);
GMAT ([targ_body '.TextureMapFileName = "C:\Users\User\Desktop\GMAT files\GenericCelestialBody.jpg"']);


%----------------------------------------
%---------- Calculated Points
%----------------------------------------

Create LibrationPoint SunJupiterL4;
GMAT SunJupiterL4.Primary = Sun;
GMAT SunJupiterL4.Secondary = Jupiter;
GMAT SunJupiterL4.Point = L4;

```

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

Create Spacecraft Mission_Spacecraft;
GMAT Mission_Spacecraft.DateFormat = UTCGregorian;
GMAT (['Mission_Spacecraft.Epoch = ''' char(start_epoch) ' 11:59:28.000'''])
GMAT Mission_Spacecraft.CoordinateSystem = SunMJ2000Eq;
GMAT Mission_Spacecraft.DisplayStateType = Cartesian;
GMAT(['Mission_Spacecraft.X = ' num2str(r0(1), '%f')]);
GMAT(['Mission_Spacecraft.Y = ' num2str(r0(2), '%f')]);
GMAT(['Mission_Spacecraft.Z = ' num2str(r0(3), '%f')]);
GMAT(['Mission_Spacecraft.VX = ' num2str(v0(1), '%f')]);
GMAT(['Mission_Spacecraft.VY = ' num2str(v0(2), '%f')]);
GMAT(['Mission_Spacecraft.VZ = ' num2str(v0(3), '%f')]);
% GMAT(['Mission_Spacecraft.CoordinateSystem = ' start_body, 'MJ2000Eq']);
% GMAT Mission_Spacecraft.DisplayStateType = Cartesian;
```

```
% GMAT Mission_Spacecraft.X = 0;
% GMAT Mission_Spacecraft.Y = 0;
% GMAT Mission_Spacecraft.Z = 0;
% GMAT(['Mission_Spacecraft.VX = ' num2str(v0_rel(1), '%f')]);
% GMAT(['Mission_Spacecraft.VY = ' num2str(v0_rel(2), '%f')]);
% GMAT(['Mission_Spacecraft.VZ = ' num2str(v0_rel(3), '%f')]);
GMAT Mission_Spacecraft.DryMass = 850;
GMAT Mission_Spacecraft.Cd = 2.2;
GMAT Mission_Spacecraft.Cr = 1.8;
GMAT Mission_Spacecraft.DragArea = 15;
GMAT Mission_Spacecraft.SRPArea = 1;
GMAT Mission_Spacecraft.NAIFId = -123456789;
GMAT Mission_Spacecraft.NAIFIdReferenceFrame = -123456789;
GMAT Mission_Spacecraft.Id = 'SatId';
GMAT Mission_Spacecraft.Attitude = CoordinateSystemFixed;
GMAT Mission_Spacecraft.ModelFile = '../data/vehicle/models/aura.3ds';
GMAT Mission_Spacecraft.ModelOffsetX = 0;
```

```
GMAT Mission_Spacecraft.ModelOffsetY = 0;
GMAT Mission_Spacecraft.ModelOffsetZ = 0;
GMAT Mission_Spacecraft.ModelRotationX = 0;
GMAT Mission_Spacecraft.ModelRotationY = 0;
GMAT Mission_Spacecraft.ModelRotationZ = 0;
GMAT Mission_Spacecraft.ModelScale = 3;
GMAT Mission_Spacecraft.AttitudeDisplayStateType = 'Quaternion';
GMAT Mission_Spacecraft.AttitudeRateDisplayStateType = 'AngularVelocity';
GMAT Mission_Spacecraft.AttitudeCoordinateSystem = 'EarthMJ2000Eq';
GMAT Mission_Spacecraft.Q1 = 0;
GMAT Mission_Spacecraft.Q2 = 0;
GMAT Mission_Spacecraft.Q3 = 0;
GMAT Mission_Spacecraft.Q4 = 1;
GMAT Mission_Spacecraft.EulerAngleSequence = '321';
GMAT Mission_Spacecraft.AngularVelocityX = 0;
GMAT Mission_Spacecraft.AngularVelocityY = 0;
GMAT Mission_Spacecraft.AngularVelocityZ = 0;
```

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

Create ForceModel DefaultProp_ForceModel;
GMAT DefaultProp_ForceModel.CentralBody = Sun;
GMAT DefaultProp_ForceModel.PointMasses = {Sun};
GMAT DefaultProp_ForceModel.Drag = None;
GMAT DefaultProp_ForceModel.SRP = On;
GMAT DefaultProp_ForceModel.RelativisticCorrection = Off;
GMAT DefaultProp_ForceModel.ErrorControl = RSSStep;
GMAT DefaultProp_ForceModel.SRP.Flux = 1367;
GMAT DefaultProp_ForceModel.SRP.Nominal_Sun = 149597870.691;

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

```
%---------- Propagators
%----------------------------------------

Create Propagator DefaultProp;
GMAT DefaultProp.FM = DefaultProp_ForceModel;
GMAT DefaultProp.Type = RungeKutta89;
GMAT DefaultProp.InitialStepSize = 60;
GMAT DefaultProp.Accuracy = 9.999999999999999e-012;
GMAT DefaultProp.MinStep = 0.001;
GMAT DefaultProp.MaxStep = 2700;
GMAT DefaultProp.MaxStepAttempts = 50;
GMAT DefaultProp.StopIfAccuracyIsViolated = true;

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

```

```
Create ImpulsiveBurn DefaultIB;
GMAT DefaultIB.CoordinateSystem = SunMJ2000Eq;
GMAT DefaultIB.Element1 = 0;
GMAT DefaultIB.Element2 = 0;
GMAT DefaultIB.Element3 = 0;
GMAT DefaultIB.DecrementMass = false;
GMAT DefaultIB.Isp = 300;
GMAT DefaultIB.GravitationalAccel = 9.810000000000001;

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

Create CoordinateSystem SunMJ2000Eq;
GMAT SunMJ2000Eq.Origin = Sun;
GMAT SunMJ2000Eq.Axes = MJ2000Eq;

```

```
Create (['CoordinateSystem ' start_body 'MJ2000Eq']);
GMAT ([start_body 'MJ2000Eq.Origin = ' start_body]);
GMAT ([start_body 'MJ2000Eq.Axes = MJ2000Eq']);
Create (['CoordinateSystem ' targ_body 'MJ2000Eq']);
GMAT ([targ_body 'MJ2000Eq.Origin = ' targ_body]);
GMAT ([targ_body 'MJ2000Eq.Axes = MJ2000Eq']);
%---------------------------------------
%---------- Solvers
%---------------------------------------
Create DifferentialCorrector DefaultDC;
GMAT DefaultDC.ShowProgress = true;
GMAT DefaultDC.ReportStyle = Normal;
```

```
GMAT DefaultDC.ReportFile = '"C:\Users\User\Desktop\GMAT files\DifferentialCorrectorDefaultDC.data';
GMAT (['DefaultDC.MaximumIterations = ' num2str(maxiter)]);
GMAT DefaultDC.DerivativeMethod = ForwardDifference;

%----------------------------------------
%---------- Subscribers
%----------------------------------------

Create OrbitView DefaultOrbitView;
GMAT DefaultOrbitView.SolverIterations = Current;
GMAT DefaultOrbitView.UpperLeft = [ -0.0100250626566416 0.04470938897168406 ];
GMAT DefaultOrbitView.Size = [ 1.020050125313283 1.056631892697467 ];
GMAT DefaultOrbitView.RelativeZOrder = 154;
GMAT (['DefaultOrbitView.Add = {Mission_Spacecraft, Earth, Jupiter, ' start_body ', ' targ_body '}']);
GMAT DefaultOrbitView.CoordinateSystem = SunMJ2000Eq;
```

```
GMAT DefaultOrbitView.DrawObject = [ true true true true true];
GMAT DefaultOrbitView.OrbitColor = [ 255 32768 1743054 16744448
16711808];
GMAT DefaultOrbitView.TargetColor = [ 4227327 0 4227327 4227327 4227327];
GMAT DefaultOrbitView.DataCollectFrequency = 1;
GMAT DefaultOrbitView.UpdatePlotFrequency = 50;
GMAT DefaultOrbitView.NumPointsToRedraw = 0;
GMAT DefaultOrbitView.ShowPlot = true;
GMAT DefaultOrbitView.ViewPointReference = Sun;
GMAT DefaultOrbitView.ViewPointVector = [ 0 0 1000000000 ];
GMAT DefaultOrbitView.ViewDirection = Mission_Spacecraft;
GMAT DefaultOrbitView.ViewScaleFactor = 1;
GMAT DefaultOrbitView.ViewUpCoordinateSystem = SunMJ2000Eq;
GMAT DefaultOrbitView.ViewUpAxis = Y;
GMAT DefaultOrbitView.CelestialPlane = Off;
GMAT DefaultOrbitView.XYPlane = On;
GMAT DefaultOrbitView.WireFrame = Off;
```

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


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

% BeginMissionSequence;
Target DefaultDC {SolveMode = Solve, ExitMode = SaveAndContinue};
    Vary (['DefaultDC(DefaultIB.Element1 = ' num2str(burn(1), '%f') ', {Perturbation = .001, Lower = -100, Upper = 100, MaxStep = .2})']);
```

```
    Vary (['DefaultDC(DefaultIB.Element2 = ' num2str(burn(2), '%f') ', {Perturbation = .001, Lower = -100, Upper = 100, MaxStep = .2})']);
    Vary (['DefaultDC(DefaultIB.Element3 = ' num2str(burn(3), '%f') ', {Perturbation = .001, Lower = -100, Upper = 100, MaxStep = .2})']);
    Maneuver DefaultIB(Mission_Spacecraft);
    Propagate (['DefaultProp(Mission_Spacecraft) {Mission_Spacecraft.ElapsedDays = ' num2str(time_inc) '}']);
    Achieve (['DefaultDC(Mission_Spacecraft.' targ_body '.RMAG = 6000, {Tolerance = 6000})']);
EndTarget;  % For targeter DefaultDC
%% Run the scenario
BuildRunGMAT();
WaitForGMAT; % Wait for GMAT to finish running
%% Get result back out
DIB = GetGMATObject('DefaultIB');
```

```
burn_soln = [
    DIB.Element1
    DIB.Element2
    DIB.Element3
    ];

sc = GetGMATObject('Mission_Spacecraft');
r_end = [
    sc.X
    sc.Y
    sc.Z
    ];
v_end = [
    sc.VX
    sc.VY
    sc.VZ
    ];
```

```
end
```

Plot_Full_Misson

The code is developed for visualization of the trajectory between the given set of four asteroids using GMAT software.

Lines 15-53 include the four Trojan asteroids Achilles, Hektor, Nestor and Agamemnon in the Sun-Jupiter L4 region. All asteroids are provided with the necessary spice kernels for information regarding space geometry and orientation data. The Sun is selected as the central body.

Lines 61-64 create the position of L4, 60˚ ahead of Jupiter. The Sun is at the center and primary while Jupiter is the secondary.

Lines 70-108 are similar to lines 50-96 of the function Trojan _distance_GMAT described above.

Lines 115-123 are similar to lines 103-111 of the function Trojan _distance_GMAT described above.

Lines 129-137 are similar to lines 117-125 of the function Trojan _distance_GMAT described above.

Lines 143-168 are similar to lines 131-138 of the function Trojan _distance_GMAT described above.

Lines 174-176, are similar to lines 144-154of the function Trojan _distance_GMAT described above.

Lines 183-188 are similar to lines 161-166 of the function Trojan _distance_GMAT described above.

Lines 194-223 are similar to lines 172-201 of the function Trojan _distance_GMAT described above.

Lines 230-236 define the mission sequence. Burn1 is executed for the propagation of the spacecraft at the starting asteroid to encounter the target asteroid within the given time. This target asteroid will be the starting asteroid for Burn2. At this point Burn2 will be executed to propagate the spacecraft to encounter the next asteroid. The above procedure is repeated for Burn3 and so on to reach the remaining asteroids. Thus, we will get the optimum trajectory through the given set of asteroids.

Lines 245-257 ultimately give the final position and velocity associated with the spacecraft in X, Y and Z direction.

Plot_Full_Mission

```
function [r_end,v_end] = Plot_Full_Mission(r0,v0,start_epoch,time_inc,burn_array)

% the inputs start_body and targ_body are strings
% The inputs r0, v0, and burn are 3x1 vectors of xyz velocities in km/s


%% Initialize GMAT
OpenGMAT();
ClearGMAT();

 %----------------------------------------
 %---------- User-Defined Celestial Bodies
 %----------------------------------------

 Create Asteroid Achilles;
 GMAT Achilles.NAIFId = 2000588;
```

```
GMAT Achilles.OrbitSpiceKernelName = {"C:\Users\User\Desktop\GMAT files\Achilles.bsp'};
GMAT Achilles.EquatorialRadius = 6378.1363;
GMAT Achilles.Flattening = 0.0033527;
GMAT Achilles.Mu = 398600.4415;
GMAT Achilles.PosVelSource = 'SPICE';
GMAT Achilles.CentralBody = 'Sun';
GMAT Achilles.TextureMapFileName = "C:\Users\User\Desktop\GMAT files\GenericCelestialBody.jpg';

Create Asteroid Hektor;
GMAT Hektor.NAIFId = 2000624;
GMAT Hektor.OrbitSpiceKernelName = {"C:\Users\User\Desktop\GMAT files\Hektor.bsp'};
GMAT Hektor.EquatorialRadius = 6378.1363;
GMAT Hektor.Flattening = 0.0033527;
GMAT Hektor.Mu = 398600.4415;
GMAT Hektor.PosVelSource = 'SPICE';
```

```
GMAT Hektor.CentralBody = 'Sun';
GMAT Hektor.TextureMapFileName = "C:\Users\User\Desktop\GMAT files\GenericCelestialBody.jpg';
Create Asteroid Nestor;
GMAT Nestor.NAIFId = 2000659;
GMAT Nestor.OrbitSpiceKernelName = {"C:\Users\User\Desktop\GMAT files\Nestor.bsp'};
GMAT Nestor.EquatorialRadius = 6378.1363;
GMAT Nestor.Flattening = 0.0033527;
GMAT Nestor.Mu = 398600.4415;
GMAT Nestor.PosVelSource = 'SPICE';
GMAT Nestor.CentralBody = 'Sun';
GMAT Nestor.TextureMapFileName = "C:\Users\User\Desktop\GMAT files\GenericCelestialBody.jpg';
Create Asteroid Agamemnon;
GMAT Agamemnon.NAIFId = 2000911;
```

```
GMAT Agamemnon.OrbitSpiceKernelName = {"C:\Users\User\Desktop\GMAT files\Agamemnon.bsp'};
GMAT Agamemnon.EquatorialRadius = 6378.1363;
GMAT Agamemnon.Flattening = 0.0033527;
GMAT Agamemnon.Mu = 398600.4415;
GMAT Agamemnon.PosVelSource = 'SPICE';
GMAT Agamemnon.CentralBody = 'Sun';
GMAT Agamemnon.TextureMapFileName = "C:\Users\User\Desktop\GMAT files\GenericCelestialBody.jpg';



%----------------------------------------
%---------- Calculated Points
%----------------------------------------

Create LibrationPoint SunJupiterL4;
```

```
GMAT SunJupiterL4.Primary = Sun;
GMAT SunJupiterL4.Secondary = Jupiter;
GMAT SunJupiterL4.Point = L4;

%--------------------------------------
%---------- Spacecraft
%--------------------------------------

Create Spacecraft Mission_Spacecraft;
GMAT Mission_Spacecraft.DateFormat = UTCGregorian;
GMAT (['Mission_Spacecraft.Epoch = ''' char(start_epoch) ' 11:59:28.000'''])
GMAT Mission_Spacecraft.CoordinateSystem = SunMJ2000Eq;
GMAT Mission_Spacecraft.DisplayStateType = Cartesian;
GMAT(['Mission_Spacecraft.X = ' num2str(r0(1), '%f')]);
GMAT(['Mission_Spacecraft.Y = ' num2str(r0(2), '%f')]);
GMAT(['Mission_Spacecraft.Z = ' num2str(r0(3), '%f')]);
GMAT(['Mission_Spacecraft.VX = ' num2str(v0(1), '%f')]);
```

```
GMAT(['Mission_Spacecraft.VY = ' num2str(v0(2), '%f')]);
GMAT(['Mission_Spacecraft.VZ = ' num2str(v0(3), '%f')]);
GMAT Mission_Spacecraft.DryMass = 850;
GMAT Mission_Spacecraft.Cd = 2.2;
GMAT Mission_Spacecraft.Cr = 1.8;
GMAT Mission_Spacecraft.DragArea = 15;
GMAT Mission_Spacecraft.SRPArea = 1;
GMAT Mission_Spacecraft.NAIFId = -123456789;
GMAT Mission_Spacecraft.NAIFIdReferenceFrame = -123456789;
GMAT Mission_Spacecraft.Id = 'SatId';
GMAT Mission_Spacecraft.Attitude = CoordinateSystemFixed;
GMAT Mission_Spacecraft.ModelFile = '../data/vehicle/models/aura.3ds';
GMAT Mission_Spacecraft.ModelOffsetX = 0;
GMAT Mission_Spacecraft.ModelOffsetY = 0;
GMAT Mission_Spacecraft.ModelOffsetZ = 0;
GMAT Mission_Spacecraft.ModelRotationX = 0;
GMAT Mission_Spacecraft.ModelRotationY = 0;
```

```
GMAT Mission_Spacecraft.ModelRotationZ = 0;
GMAT Mission_Spacecraft.ModelScale = 3;
GMAT Mission_Spacecraft.AttitudeDisplayStateType = 'Quaternion';
GMAT Mission_Spacecraft.AttitudeRateDisplayStateType = 'AngularVelocity';
GMAT Mission_Spacecraft.AttitudeCoordinateSystem = 'EarthMJ2000Eq';
GMAT Mission_Spacecraft.Q1 = 0;
GMAT Mission_Spacecraft.Q2 = 0;
GMAT Mission_Spacecraft.Q3 = 0;
GMAT Mission_Spacecraft.Q4 = 1;
GMAT Mission_Spacecraft.EulerAngleSequence = '321';
GMAT Mission_Spacecraft.AngularVelocityX = 0;
GMAT Mission_Spacecraft.AngularVelocityY = 0;
GMAT Mission_Spacecraft.AngularVelocityZ = 0;
%----------------------------------------
%---------- ForceModels
```

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

Create ForceModel DefaultProp_ForceModel;
GMAT DefaultProp_ForceModel.CentralBody = Sun;
GMAT DefaultProp_ForceModel.PointMasses = {Sun};
GMAT DefaultProp_ForceModel.Drag = None;
GMAT DefaultProp_ForceModel.SRP = On;
GMAT DefaultProp_ForceModel.RelativisticCorrection = Off;
GMAT DefaultProp_ForceModel.ErrorControl = RSSStep;
GMAT DefaultProp_ForceModel.SRP.Flux = 1367;
GMAT DefaultProp_ForceModel.SRP.Nominal_Sun = 149597870.691;

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

Create Propagator DefaultProp;
```

```
GMAT DefaultProp.FM = DefaultProp_ForceModel;
GMAT DefaultProp.Type = RungeKutta89;
GMAT DefaultProp.InitialStepSize = 60;
GMAT DefaultProp.Accuracy = 9.999999999999999e-012;
GMAT DefaultProp.MinStep = 0.001;
GMAT DefaultProp.MaxStep = 2700;
GMAT DefaultProp.MaxStepAttempts = 50;
GMAT DefaultProp.StopIfAccuracyIsViolated = true;

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

Create ImpulsiveBurn Burn1;
GMAT Burn1.CoordinateSystem = SunMJ2000Eq;
GMAT (['Burn1.Element1 = ' num2str(burn_array(1,1), '%f')]);
GMAT (['Burn1.Element2 = ' num2str(burn_array(2,1), '%f')]);
```

```
GMAT (['Burn1.Element3 = ' num2str(burn_array(3,1), '%f')]);
GMAT Burn1.DecrementMass = false;
GMAT Burn1.Isp = 300;
GMAT Burn1.GravitationalAccel = 9.810000000000001;
Create ImpulsiveBurn Burn2;
GMAT Burn2.CoordinateSystem = SunMJ2000Eq;
GMAT (['Burn2.Element1 = ' num2str(burn_array(1,2), '%f')]);
GMAT (['Burn2.Element2 = ' num2str(burn_array(2,2), '%f')]);
GMAT (['Burn2.Element3 = ' num2str(burn_array(3,2), '%f')]);
GMAT Burn2.DecrementMass = false;
GMAT Burn2.Isp = 300;
GMAT Burn2.GravitationalAccel = 9.810000000000001;
Create ImpulsiveBurn Burn3;
GMAT Burn3.CoordinateSystem = SunMJ2000Eq;
GMAT (['Burn3.Element1 = ' num2str(burn_array(1,3), '%f')]);
```

```
GMAT (['Burn3.Element2 = ' num2str(burn_array(2,3), '%f')]);
GMAT (['Burn3.Element3 = ' num2str(burn_array(3,3), '%f')]);
GMAT Burn3.DecrementMass = false;
GMAT Burn3.Isp = 300;
GMAT Burn3.GravitationalAccel = 9.810000000000001;

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

Create CoordinateSystem SunMJ2000Eq;
GMAT SunMJ2000Eq.Origin = Sun;
GMAT SunMJ2000Eq.Axes = MJ2000Eq;


%----------------------------------------
%---------- Solvers
```

```
%----------------------------------------
% Create DifferentialCorrector DefaultDC;
% GMAT DefaultDC.ShowProgress = true;
% GMAT DefaultDC.ReportStyle = Normal;
% GMAT DefaultDC.ReportFile = "C:\Users\User\Desktop\GMAT files\DifferentialCorrectorDefaultDC.data';
% GMAT (['DefaultDC.MaximumIterations = ' num2str(maxiter)]);
% GMAT DefaultDC.DerivativeMethod = ForwardDifference;
%----------------------------------------
%---------- Subscribers
%----------------------------------------
Create OrbitView DefaultOrbitView;
GMAT DefaultOrbitView.SolverIterations = Current;
```

```
GMAT DefaultOrbitView.UpperLeft = [ -0.0100250626566416 -0.04470938897168406 ];
GMAT DefaultOrbitView.Size = [ 1.020050125313283 1.056631892697467 ];
GMAT DefaultOrbitView.RelativeZOrder = 154;
GMAT DefaultOrbitView.Add = {Mission_Spacecraft, Earth, Jupiter, Achilles, Hektor, Nestor, Agamemnon};
GMAT DefaultOrbitView.CoordinateSystem = SunMJ2000Eq;
GMAT DefaultOrbitView.DrawObject = [ true true true true true true true];
GMAT DefaultOrbitView.OrbitColor = [ 255 32768 1743054 16744448 16711808 65535 8388863 ];
GMAT DefaultOrbitView.TargetColor = [ 4227327 0 4227327 4227327 4227327 4227327 4227327];
GMAT DefaultOrbitView.DataCollectFrequency = 1;
GMAT DefaultOrbitView.UpdatePlotFrequency = 50;
GMAT DefaultOrbitView.NumPointsToRedraw = 0;
GMAT DefaultOrbitView.ShowPlot = true;
GMAT DefaultOrbitView.ViewPointReference = Sun;
GMAT DefaultOrbitView.ViewPointVector = [ 0 0 1000000000 ];
```

```
GMAT DefaultOrbitView.ViewDirection = Mission_Spacecraft;
GMAT DefaultOrbitView.ViewScaleFactor = 1;
GMAT DefaultOrbitView.ViewUpCoordinateSystem = SunMJ2000Eq;
GMAT DefaultOrbitView.ViewUpAxis = Y;
GMAT DefaultOrbitView.CelestialPlane = Off;
GMAT DefaultOrbitView.XYPlane = On;
GMAT DefaultOrbitView.WireFrame = Off;
GMAT DefaultOrbitView.Axes = On;
GMAT DefaultOrbitView.Grid = Off;
GMAT DefaultOrbitView.SunLine = Off;
GMAT DefaultOrbitView.UseInitialView = On;
GMAT DefaultOrbitView.StarCount = 7000;
GMAT DefaultOrbitView.EnableStars = On;
GMAT DefaultOrbitView.EnableConstellations = Off;


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

```
%---------- Mission Sequence
%----------------------------------------
% BeginMissionSequence;
Maneuver Burn1(Mission_Spacecraft);
Propagate (['DefaultProp(Mission_Spacecraft) {Mission_Spacecraft.ElapsedDays = ' num2str(time_inc) '}']);
Maneuver Burn2(Mission_Spacecraft);
Propagate (['DefaultProp(Mission_Spacecraft) {Mission_Spacecraft.ElapsedDays = ' num2str(time_inc) '}']);
Maneuver Burn3(Mission_Spacecraft);
Propagate (['DefaultProp(Mission_Spacecraft) {Mission_Spacecraft.ElapsedDays = ' num2str(time_inc) '}']);


%% Run the scenario
BuildRunGMAT();
WaitForGMAT; % Wait for GMAT to finish running
```

```
%% Get result back out

sc = GetGMATObject('Mission_Spacecraft');
r_end = [
    sc.X
    sc.Y
    sc.Z
    ];
v_end = [
    sc.VX
    sc.VY
    sc.VZ
    ];

end
```