Trajectory Control and Optimization for Responsive Spacecraft

THESIS

Costantinos Zagaris, Captain, USAF

AFIT/GA/ENY/12-M13

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GA/ENY/12-M13

Trajectory Control and Optimization for Responsive Spacecraft

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Astronautical Engineering

Costantinos Zagaris, B.S. Aerospace Engineering

Captain, USAF

March 2012

AFIT/GA/ENY/12-M13

Trajectory Control and Optimization for Responsive Spacecraft

Costantinos Zagaris, B.S. Aerospace Engineering
Captain, USAF

Approved:

Jonathan T. Black, PhD (Chairman)	Date
Ronald J. Simmons, LtCol, USAF (Member)	Date
Nathan A. Titus, PhD (Member)	Date

Abstract

The concept of responsive space has been gaining interest, and growing to include systems that can be re-tasked to complete multiple missions within their lifetime. The purpose of this study is to develop an algorithm that produces a maneuver trajectory that will cause a spacecraft to arrive at a particular location within its orbit earlier than expected. The time difference, Δt, is used as a metric to quantify the effects of the maneuver. Two separate algorithms are developed. The first algorithm is an optimal control method and is developed through Optimal Control Theory. The second algorithm is a feedback control method and is developed through Lyapunov Theory. It is shown that the two algorithms produce equivalent results for the maneuvers discussed. In-plane maneuver results are analyzed analytically, and an algebraic expression for Δt is derived. Examples are provided of how the analytic expression can be used for mission planning purposes. The feedback control algorithm is then further developed to demonstrate the simplicity of implementing additional capabilities. Finally, a set of simulations is analyzed to show that in order to maximize the amount of Δt achieved, a spacecraft must be allowed as much lead time as possible, and begin thrusting as early as possible.

Acknowledgements

I believe that one of the most important things in life is recognizing those around you that play a significant role in achieving your goals. Without a doubt, if I didn't have the support of my professors, family, and friends, I would not have been able to complete my research and keep my sanity. First, I would like to thank my advisor, Dr. Black, for motivating me to solve my problems. There were many times that I turned to him for advice when things were not working as they should, and every time he guided me in the direction that would lead me to the answer I was looking for.

I would like to thank the members of my committee for taking the time to give me valuable feedback throughout my research. Also, thanks to all my professors at AFIT for providing me with the necessary knowledge and tools to excel, and for helping me both inside and out of the classroom. I always felt comfortable turning to my professors for help, even if they were not directly involved in my research.

Lastly, and most importantly, I would like to thank my wife, and best friend, for always believing in me and having the patience to listen to me vent about schoolwork. Her unwavering support throughout my entire career always reminds me how lucky I am to have her.

Costantinos Zagaris

Table of Contents

Page

List of Figures

List of Tables

List of Symbols

Roman

$\boldsymbol{A}$	Perturbing acceleration vector
a_r	Perturbing acceleration component in the radial direction
a_θ	Perturbing acceleration component in the transverse direction
a_h	Perturbing acceleration component in the orbit-normal direction
a	Semi-major axis
e	Eccentricity
E	Eccentric anomaly
F	Eccentric longitude
$\boldsymbol{F}_g$	Force due to gravity
$[\hat{f}, \hat{g}, \hat{w}]$	Unit vectors making up the Equinoctial frame
G	Universal gravitational constant
g_0	Gravitational acceleration at sea level
H	Hamiltonian
h	Eccentricity vector component in the $\hat{f}$ direction
I_{sp}	Engine Specific Impulse
i	Inclination
$[\hat{i}, \hat{j}, \hat{k}]$	Unit vectors making up the Earth-Centered-Inertial frame
$[\hat{i}_r, \hat{i}_\theta, \hat{i}_h]$	Unit vectors making up the Local-Vertical-Local-Horizontal frame
J	Cost functional
k	Eccentricity vector component in the $\hat{g}$ direction
L	True longitude
M	Mean anomaly
m	Spacecraft mass
n	Mean motion
p	Equinoctial element
q	Equinoctial element
$\boldsymbol{r}$	Inertial position vector
r	Position vector magnitude
T	Constant thrust magnitude
T_{on}	Duration of pulse
T_{off}	Duration of coasting period between pulses
t	Time
$\boldsymbol{u}$	Control vector
$\boldsymbol{v}$	Inertial velocity vector
V	Velocity vector magnitude
v_x	Inertial velocity component along the $\hat{i}$ vector
v_y	Inertial velocity component along the $\hat{i}$ vector
v_z	Inertial velocity component along the $\hat{i}$ vector
v_r	Velocity component in the radial direction
v_s	Velocity component in the transverse direction

$\boldsymbol{x}$	State vector
x	Inertial position component along the $\hat{i}$ vector
y	Inertial position component along the $\hat{j}$ vector
z	Inertial position component along the $\hat{k}$ vector

Greek

α	Control weighting factor
β	Out-of-plane control angle
γ	Greenwich local sidereal time
γ_g	Greenwich sidereal time at epoch
$\Delta()$	Change in a certain variable
$\delta()$	Variation of a certain variable
ϵ	Specific mechanical energy
θ	In-plane control angle
λ	Mean longitude
λ	Lagrange multiplier or costate
μ	Earth gravitational constant
ν	True anomaly
ϕ	Control angle in polar coordinates
ψ	Terminal state constraint function
Ω	Right ascension of the ascending node
ω	Argument of perigee
ω_e	Earth's rotation rate

Trajectory Control and Optimization for Responsive Spacecraft

I. Introduction

Over the past few years interest in responsive space systems has become more prevalent. The Operationally Responsive Space (ORS) office has been tasked with the responsibility of rapidly developing new capabilities to provide low-cost access to space. According to Col Newberry, however, the term "responsive space" defines a system that is not only developed quickly but could also respond to new taskings within days, hours, or even minutes [1]. In order to achieve true responsiveness at the lowest cost, spacecraft must be able to maneuver multiple times throughout their lifetime and accomplish multiple missions. Allowing space assets to be re-tasked in this manner could significantly decrease the overall cost of maintaining space programs. Dr. Wertz writes that the average cost of a space system launched by the United States is 2.5 billion dollars, and average development time approaching a decade [2]. The goal for responsive systems is to supplement on-orbit assets in order to drive the cost down to less than 20 million dollars [2].

Newberry discusses responsive maneuvers that will place a satellite over a specific geographical location on Earth within a specific time, and in an unwarned manner [1]. Co *et al* investigate the use of both impulsive thrust, chemical propulsion (CP), and low thrust, electric propulsion (EP), systems to maneuver a spacecraft in Low-Earth Orbit (LEO) and achieve operational responsiveness by changing its ground track [3]. They analyze the maneuvers by comparing the ground track of the maneuvering spacecraft to the ground track of a non-maneuvering reference spacecraft with the same initial conditions. The metric they used to quantify the effects of the maneuvers was the terrestrial distance between the maneuvering and reference spacecraft [3]. Their work demonstrated that Newberry's concept of operational re-

sponsiveness is indeed achievable, but they did not address the possibility of arriving on target in an unwarned manner.

The Joint Space Operations Center (JSpOC), under US Strategic Command, has the responsibility of detecting, tracking, and identifying all man-made objects in Earth orbit. The JSpOC tracks more than 22,000 objects and updates their positions in order to keep the Space Catalog current [4]. With that many objects being tracked it is conceivable that a spacecraft could change its orbit just enough to be confused with a different object, especially if the spacecraft maneuver is not detected. Large thrust, impulsive maneuvers, are relatively easy to detect. However, maneuvers using small accelerations can be hard to discriminate from other natural perturbations [5]. The maneuver detection and orbit estimation problem for a low-thrust maneuvering spacecraft is an active research topic, and is not in the scope of this thesis. Nevertheless, it is important to have a basic understanding of orbit determination in order to understand how a spacecraft's orbit estimate can become degraded.

The objective of orbit determination is to estimate current and future states of an orbiting spacecraft subject to non-deterministic dynamics. The full-scope orbit determination problem would include using estimation theory to compute covariance ellipsoids at beginning and final times. Estimation theory provides many different methods that can be used in orbit determination the earliest of which is the Least Squares method discovered by Gauss, which assumes that the dynamics of the system are deterministic [6]. Orbit determination methods that include stochastic dynamical systems were later developed and resulted in the invention of the Kalman filter and the Bayes filter [6]. Many variations to the Kalman filter also exist such as the Extended Kalman filter, Unscented Kalman filter, and Backwards Smoothing Kalman filter [7]. The list of existing estimation algorithms is endless, and modern estimation techniques are still popular research topics.

Most current research in the topic of orbit determination is motivated by the need to increase situational awareness in space. Concerns such as detecting large

and small maneuvers, and providing better estimates are addressed. The goal of this research, however, is to provide responsive flexible collection based on user taskings. In his thesis, Payte compared the performance of current space surveillance capabilities to the proposed *Space Fence* system [8]. He used Monte-Carlo methods to incorporate inherent radar observation errors, and propagated the orbit from an initial acquisition time to the re-acquisition time. His simulation showed that the most significant factor in degrading the reconstruction of the spacecraft's orbit is the time between initial acquisitions. A smaller amount of time between initial acquisitions resulted in a more uncertain orbit estimate. It then follows that if a spacecraft performed a maneuver to change its arrival time over a particular location, therefore reducing the re-acquisition time, flexible collection could be achievable.

The research discussed in this thesis somewhat parallels the work done by Co *et al* in [3] but time is used as a metric instead of terrestrial distance. Although both CP and EP systems could be used to perform these maneuvers, EP systems are appealing due to their high efficiency and lower weight [2]. The small thrust output by an EP system perturbs the spacecraft's orbit slowly and maneuver durations will be longer. These perturbations, however, could be a key factor in responsive flexible collection. Therefore, only maneuvers using EP systems are considered throughout this research. EP has been used extensively in space and the technology is well developed. Current systems in space use EP for high-precision station-keeping or orbit maintenance maneuvers. The tradition of using CP for larger maneuvers has slowed the realization of the advantage of EP systems, but the potential is undoubtedly present [9]. In the context of Newberry's concept, EP could be used to perform in-plane and out-of-plane maneuvers to change the spacecraft's arrival time over a specified target.

1.1 Problem Statement

The objective of this research is to develop an algorithm that produces a trajectory for responsive spacecraft. The computed trajectory is compared to a reference,

non-maneuvering, trajectory and the time difference, Δt, at a specified location within the orbit is used as the metric to quantify the effects of the maneuver. The study is divided into three phases - maneuvers using optimal control techniques, maneuvers using feedback control techniques, and scenario simulations.

1.1.1 Phase One: Optimal Control Methods for Responsive Maneuvers. This phase focuses on methods for performing minimum time in-plane maneuvers. The optimal control problem for an orbiting spacecraft perturbed by a small, constant acceleration is formulated using Euler-Lagrange theory, and solved numerically via pseudospectral methods. The magnitude of the acceleration is assumed constant, while its direction is treated as the control variable. With a constant acceleration, and no throttling, the mass flow rate of the propulsion system will be constant, therefore minimizing the duration of the maneuver will also minimize the amount of fuel used [10]. Optimal acceleration profiles for the in-plane plane problem are presented and discussed.

1.1.2 Phase Two: Feedback Control Methods for Responsive Maneuvers. This phase focuses on feedback control methods for performing in-plane maneuvers. Lyapunov theory is applied to the nonlinear equations of motion for an orbiting spacecraft perturbed by a small, constant acceleration. Results for the in-plane problem are compared to the optimal acceleration profiles computed in Phase One, and out-of-plane maneuvering capability is discussed and demonstrated. The implementation of a thrust-coast duty cycle is demonstrated to provide flexibility in applying the algorithm to several types of spacecraft that may not have the ability of thrusting continuously.

1.1.3 Phase Three: Scenario Simulation. The last phase of this research is a scenario simulation using the algorithms developed throughout this research. The scenario involves providing the spacecraft a certain amount of time to complete an in-plane maneuver that creates a specified amount of Δt.

1.2 Thesis Outline

Chapter II provides background information applicable to this research, starting by describing the equations of motion for a spacecraft experiencing a small, finite, constant acceleration. A brief background of optimal spacecraft maneuvers is presented, including the derivation of the Euler-Lagrange equations, followed by a brief background of Lyapunov Theory. Chapter III provides a detailed explanation of the models and methods that were used. Chapter IV presents the results of this research. Finally, Chapter V is a summary of the conclusions made throughout the research, and provides recommendation for future work.

II. Background

2.1 Equations of Motion

The equations of motion for a spacecraft under constant low-thrust can be written from Newton's second law. There are a number of different variables that can be used to express the motion of the spacecraft such as inertial position and velocity, classical orbital elements, and equinoctial elements. These methods are detailed in the following sections.

2.1.1 Inertial Position and Velocity. Using an inertially fixed, right-handed coordinate system is one the easiest ways to express the spacecraft's motion. A common reference frame of choice is the Earth-Centered-Inertial (ECI) frame. The three unit vectors that make up the ECI frame are $[\hat{i}, \hat{j}, \hat{k}]$ and they are oriented such that $\hat{i}$ points towards the vernal equinox, $\hat{k}$ points to the north pole, and $\hat{j}$ completes the right-handed set. Figure 2.1 is a depiction of the spacecraft with respect to the ECI frame, where $\boldsymbol{r}$ is the inertial position vector comprised of three components $[x, y, z]$, $\boldsymbol{v}$ is the inertial velocity vector comprised of three components $[v_x, v_y, v_z]$, and $\boldsymbol{A}$ is the disturbance acceleration vector expessed in the ECI frame. Based on Newton's Law of Gravitation, the force due to gravity acting on the spacecraft is [11]:

$$\boldsymbol{F}_g = -\frac{Gm_e m}{r^3}\boldsymbol{r} \tag{2.1}$$

where G is the universal gravitational constant, m_e is the mass of the Earth, and m is the mass of the spacecraft. The force in equation 2.1 results in an acceleration due to gravity, referred to as the basic two-body acceleration [11]. Therefore, without the disturbance acceleration, $\boldsymbol{A}$, acting on the spacecraft, and with the fundamental assumption that $m_e >> m$, the problem would reduce to the basic two-body equation of motion:

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{r^3}\boldsymbol{r} \tag{2.2}$$

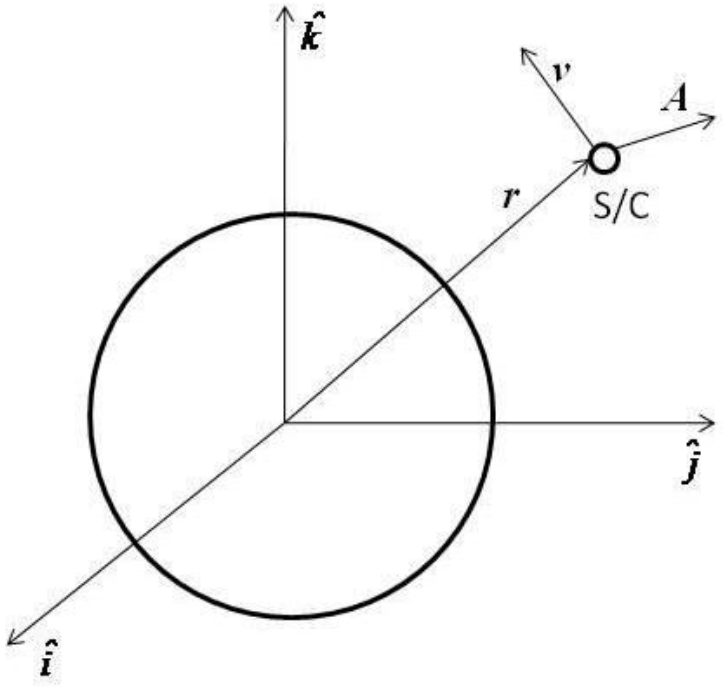

Figure 2.1: Earth Centered Inertial Reference Frame

where $\mu \approx Gm_e$ is the gravitational constant for Earth. With the disturbance acceleration present, the equation of motion simply becomes:

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{A} \tag{2.3}$$

The disturbance acceleration vector can be written as a function of an acceleration magnitude, and two angles, θ and β, that define the direction of the vector. The angle θ is measured from the $\hat{i}$ unit vector and lies on the $\hat{i}$-$\hat{j}$ plane, and β measures the angle between the acceleration vector and the $\hat{i}$-$\hat{j}$ plane. The acceleration magnitude of the vehicle is a function of the constant thrust magnitude, T, the initial mass, m_0, and the mass flow rate of propellant, $\dot{m}$ [10]:

$$A = \frac{T}{m_0 + \dot{m}t} \tag{2.4}$$

Assuming that the change in mass of the vehicle in negligible ($\dot{m} = 0$), the acceleration magnitude is a constant. Thorne used the same formulation which results in the following equations of motion [10]:

$$\dot{x} = v_x \tag{2.5}$$

$$\dot{y} = v_y \tag{2.6}$$

$$\dot{z} = v_z \tag{2.7}$$

$$\dot{v}_x = -\frac{\mu}{r^3}x + A\cos\beta\cos\theta \tag{2.8}$$

$$\dot{v}_y = -\frac{\mu}{r^3}y + A\cos\beta\sin\theta \tag{2.9}$$

$$\dot{v}_z = -\frac{\mu}{r^3}z + A\sin\beta \tag{2.10}$$

where $r = \sqrt{x^2 + y^2 + z^2}$. The six differential equations, 2.5-2.10 describe the three-dimensional motion of the spacecraft in cartesian coordiantes. Thorne simplified the problem to two dimensions by aligning the $\hat{i}$-$\hat{j}$ plane with the orbital plane, making $\hat{k}$ parallel to the orbit angular momentum vector. He also set $\beta = 0$ and $z = 0$ to set up the equations for coplanar maneuvers [10]. The six equations reduce to four coplanar equations of motion:

$$\dot{x} = v_x \tag{2.11}$$

$$\dot{y} = v_y \tag{2.12}$$

$$\dot{v}_x = -\frac{\mu}{r^3}x + A\cos\theta \tag{2.13}$$

$$\dot{v}_y = -\frac{\mu}{r^3}y + A\sin\theta \tag{2.14}$$

The coplanar equations of motion, equations 2.11-2.14, can also be expressed in polar coordinates. The position vector only has one component, r, in the radial direction, the velocity vector has two components, v_r in the radial direction, and v_s in the transverse direction. The control angle, ϕ, can be used to define the thrust angle and is measured from the velocity vector to the acceleration vector. The polar

equations of motion can be written [11]:

$$\dot{r} = v_r \tag{2.15}$$

$$\dot{v}_r = \frac{v_s^2}{r} - \frac{\mu}{r^2} + A\sin\phi \tag{2.16}$$

$$\dot{v}_s = -\frac{v_r v_s}{r} + A\cos\phi \tag{2.17}$$

Optimal control methods can be used with any of the above representations of the spacecraft dynamics to determine an optimal acceleration profile. The simplistic form of these dynamic equations is useful when attempting to come up with analytical solutions, but can cause problems when using numerical methods as described in section 2.2.2. Numerical solvers tend to be more effective when the state variables change slowly. In the cartesian formulation, the inertial position and velocity of the spacecraft changes drastically over small periods of time. Rapid changes in the state variables can cause numerical solvers to be ineffective especially when considering maneuvers over long periods of time. [12] The polar formulation, however, includes variables that change slowly as long as the acceleration magnitude remains small.

Classical perturbation theory provides a means for developing differential equations that describe the dynamic behavior of the orbital elements over time. Since the applied acceleration is very small the effect will be an osculating orbit where five of the six orbital elements change slowly with time. The sixth element, defining the spacecraft's position within the orbit, changes rapidly but can be predicted by the mean motion of the osculating orbit. Therefore, perturbation equations can be very effective in solving optimal control problems, especially when using numerical methods.

2.1.2 Classical Orbital Elements. In deriving the differential equations that describe the behavior of the orbital elements over time, Wiesel used a method referred to as the variation of paramenters (VOP) [13]. Using the known relationships between the six classical orbital elements (COEs), $(a, e, i, \Omega, \omega, \nu)$, and the position

and velocity vectors, the cartesian coordinate componets in equation 2.3 can be replaced with the COEs. The COEs define the orbit's semi-major axis, eccentricity, inclination, right ascension of the ascending node (RAAN), argument of perigee, and true anomaly respectively. If only the two-body acceleration is considered the COEs will be constant. With a perturbing acceleration present any change in the COEs will be due to that force alone. A new rotating coordinate frame is introduced, often referred to as the local-vertical-local-horizontal (LVLH) frame, whose basis is defined by the three unit vectors $[\hat{i}_r, \hat{i}_\theta, \hat{i}_h]$. The LVLH frame is attached to the spacecraft and is oriented such that $\hat{i}_r$ points along the direction of the radius vector, $\hat{i}_h$ points along the direction of the orbit's angular momentum vector, and $\hat{i}_\theta$ completes the right-handed set. It is important to note that for circular orbits $\hat{i}_\theta$ is always aligned with the spacecraft's velocity vector, while for elliptic orbits $\hat{i}_\theta$ is aligned with the velocity vector only at perigee and apogee. Figure 2.2 shows a depiction of the COEs [14] with respect to the inertial frame and the LVLH frame attached to the spacecraft.

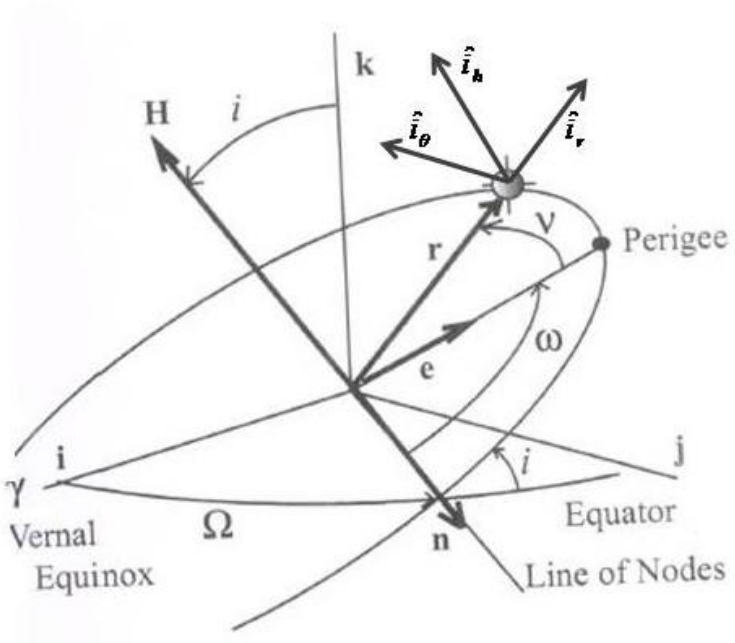

Figure 2.2: Classical Orbital Elements and Local-Vertical-Local-Horizontal Frame

A state vector containing the six COEs is defined, $\boldsymbol{e} = [a, e, i, \Omega, \omega, \nu]^T$. Since the COEs can be written in terms of the instanteneous position and velocity vectors, the new state vector $\boldsymbol{e}$ can be written as a function of position and velocity. The rate of change of $\boldsymbol{e}$ can then be written by the chain rule using the equations of motion $\dot{\boldsymbol{r}} = \boldsymbol{v}$ and $\dot{\boldsymbol{v}} = -\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{A}$ [13]:

$$\frac{d\boldsymbol{e}(\boldsymbol{r}, \boldsymbol{v})}{dt} = \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{r}} \cdot \boldsymbol{v} + \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{v}} \cdot \left(-\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{A}\right) = \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{v}} \cdot \boldsymbol{A} \tag{2.18}$$

Equation 2.18 shows that the time rate of change of the COEs due to the two-body acceleration is zero and only the perturbing acceleration contributes to the change, while not accounting for the change in ν due to the mean motion. The perturbing acceleration and the spacecraft's velocity can be written in the LVLH frame:

$$\boldsymbol{A} = a_r \hat{i}_r + a_\theta \hat{i}_\theta + a_h \hat{i}_h \tag{2.19}$$

$$\boldsymbol{v} = \dot{r}\hat{i}_r + r\dot{\nu}\hat{i}_\theta \tag{2.20}$$

The equation for the specific mechanical energy of the orbit is defined by [13]:

$$\epsilon = \frac{1}{2}\boldsymbol{v} \cdot \boldsymbol{v} - \frac{\mu}{r} \tag{2.21}$$

Taking the time derivative of equation 2.21 and substituting $\dot{\boldsymbol{v}} = \boldsymbol{A}$, so that the only change is attributed to the perturbing acceleration, yields:

$$\frac{d\epsilon}{dt} = \boldsymbol{v} \cdot \boldsymbol{A} \tag{2.22}$$

The semi-major axis of the orbit is $a = -\mu/2\epsilon$ and its time derivative is:

$$\frac{da}{dt} = \frac{2a^2}{\mu}\frac{d\epsilon}{dt} = \frac{2a^2}{\mu}(\dot{r}a_r + r\dot{\nu}a_\theta) \tag{2.23}$$

The following two-body results can then be substituted into equation 2.23:

$$\frac{dr}{d\nu} = \frac{re\sin\nu}{1+e\cos\nu} \tag{2.24}$$

$$\frac{d\nu}{dt} = \frac{na^2}{r^2}\sqrt{1-e^2} \tag{2.25}$$

$$\dot{r} = \frac{dr}{d\nu}\dot{\nu} \tag{2.26}$$

where $n = \sqrt{\mu/a^3}$ defines the mean motion of the orbit. The combined result yields the first of six Gauss' Variational Equations, also known as the acceleration form of the Lagrange Planetary Equations (LPEs):

$$\frac{da}{dt} = \frac{2e\sin\nu}{n\sqrt{1-e^2}}a_r + \frac{2a\sqrt{1-e^2}}{nr}a_\theta \tag{2.27}$$

In a similar manner, the differential equations for the other five COEs can be derived as shown by Wiesel [13]. The final LPEs are shown here using slightly different notation, but derived in the same manner by Schaub and Junkins [15].

$$\frac{da}{dt} = \frac{2a^2}{h}(e\sin\nu a_r + \frac{p}{r}a_\theta) \tag{2.28}$$

$$\frac{de}{dt} = \frac{1}{h}(p\sin\nu a_r + ((p+r)\cos\nu + re)a_\theta) \tag{2.29}$$

$$\frac{di}{dt} = \frac{r\cos(\omega+\nu)}{h}a_h \tag{2.30}$$

$$\frac{d\Omega}{dt} = \frac{r\sin(\omega+\nu)}{h\sin i}a_h \tag{2.31}$$

$$\frac{d\omega}{dt} = \frac{1}{he}(-p\cos\nu a_r + (p+r)\sin\nu a_\theta) - \frac{r\sin(\omega+\nu)\cos i}{h\sin i}a_h \tag{2.32}$$

$$\frac{d\nu}{dt} = \frac{h}{r^2} + \frac{1}{he}(p\cos\nu a_r - (p+r)\sin\nu a_\theta) \tag{2.33}$$

where $h = \sqrt{\mu p}$ is the orbit angular momentum, and $p = a(1-e^2)$ is the semi-latus rectum.

Equations 2.28-2.33 represent another set of first-order differential equations that can be used in an optimization problem. These equations are more complicated

than equations 2.5-2.10 but can be a powerful tool for numerical methods since they change slowly with time. One of the disadvantages of these equations is that they are obviously singular for circular orbits ($e = 0$) and equitorial orbits ($i = 0$). These equations could still provide useful results as long as the singularities are avoided throughout the trajectory. However, if the singularities are unavoidable equinoctial orbital elements could be used.

2.1.3 Equinoctial Elements. Equinoctial elements offer the same advantages as COEs while avoiding the singularities that appear in the LPEs. Since the relationships between the COEs and equinoctial elements are known, a transformation of variables could be applied to derive the rate of change of the equinoctial elements due to the perturbing acceleration. One of the most popular articles on the subject of equinoctial elements is by Broucke and Cefola [16]. They developed the Lagrange and Poisson brackets in terms of the equinoctial elements for applications in perturbation analysis. Their work has been applied to optimal transfers by many authors, however the discussion below is taken from Kechichian [17] and Chobotov [18].

First, the equinoctial reference frame comprised of unit vectors $[\hat{f}, \hat{g}, \hat{w}]$ is introduced. The equinoctial frame is oriented such that $\hat{f}$ and $\hat{g}$ span the orbital plane, and $\hat{w}$ is aligned with the orbit angular momentum vector. Figure 2.3 [17] shows the orientation of the equinoctial frame with respect to the ECI frame. The equinoctial frame and the ECI frame can be related by three rotations through the angles Ω, $-i$, and $-\Omega$. The resulting rotation matrix shown in equation 2.34 can be used to transform between the two frames. [18]

$$\begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix} = \begin{bmatrix} \cos^2\Omega + \cos i \sin^2\Omega & \cos\Omega\sin\Omega - \sin\Omega\cos i\cos\Omega & \sin\Omega\sin i \\ \sin\Omega\cos\Omega - \sin\Omega\cos\Omega\cos i & \sin^2\Omega + \cos i\cos^2\Omega & -\cos\Omega\sin i \\ -\sin\Omega\sin i & \sin i\cos\Omega & \cos i \end{bmatrix} \begin{bmatrix} \hat{f} \\ \hat{g} \\ \hat{w} \end{bmatrix} \tag{2.34}$$

Based on the transformation in equation 2.34 and after some algebraic manipulation the $\hat{f}$, $\hat{g}$, and $\hat{w}$ unit vectors can be expressed in the ECI frame as functions of the

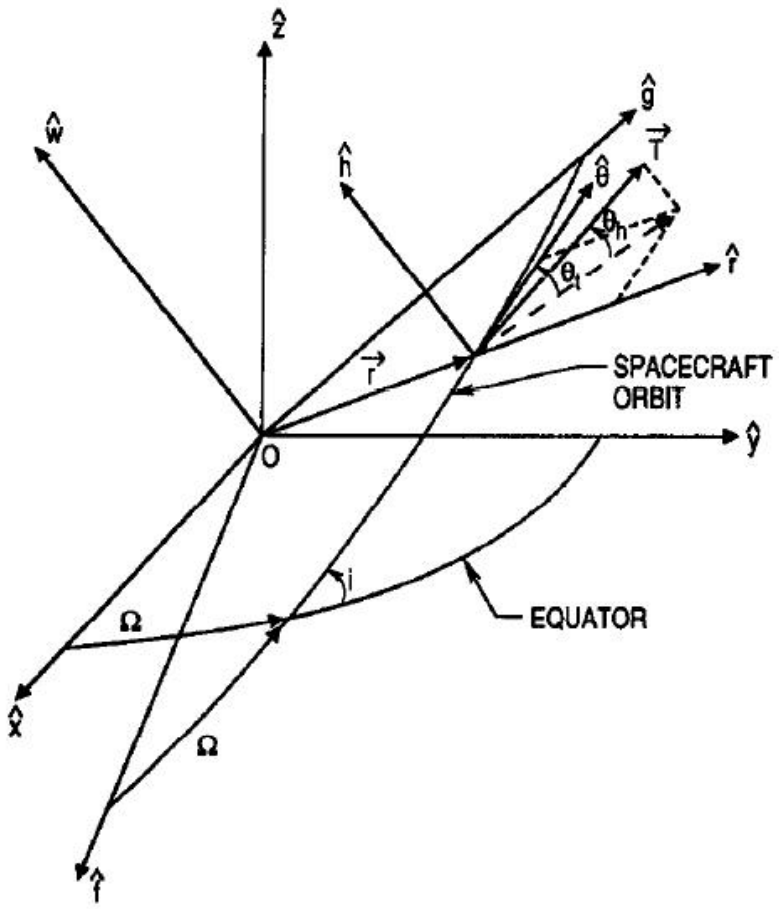

Figure 2.3: Equinoctial Frame with respect to ECI Frame [17]

equinoctial elements p and q. [18]

$$\hat{f} = \frac{1}{1+p^2+q^2}\begin{bmatrix} 1-p^2+q^2 \\ 2pq \\ -2p \end{bmatrix} \tag{2.35}$$

$$\hat{g} = \frac{1}{1+p^2+q^2}\begin{bmatrix} 2pq \\ 1+p^2-q^2 \\ 2q \end{bmatrix} \tag{2.36}$$

$$\hat{w} = \frac{1}{1+p^2+q^2}\begin{bmatrix} 2p \\ -2q \\ 1-p^2-q^2 \end{bmatrix} \tag{2.37}$$

The equinoctal element set, like the COEs, is comprised of six elements. The first element is the semimajor axis and is the same as the semi-major axis in the COEs. The elements h and k define the eccentricity vector components in the $\hat{f}$ and $\hat{g}$ directions, the elements p and q are a function of the inclination and the RAAN, and λ is the mean longitude. The equinoctial elements can be written in terms of the COEs [18]:

$$a = a \tag{2.38}$$

$$h = e\sin(\omega + \Omega) \tag{2.39}$$

$$k = e\cos(\omega + \Omega) \tag{2.40}$$

$$p = \tan\left(\frac{i}{2}\right)\sin\Omega \tag{2.41}$$

$$q = \tan\left(\frac{i}{2}\right)\cos\Omega \tag{2.42}$$

$$\lambda = M + \omega + \Omega \tag{2.43}$$

The mean longitude can be replaced by F, the eccentric longitude, or L, the true longitude, through Kepler's equation.

$$F = E + \omega + \Omega \tag{2.44}$$

$$L = \nu + \omega + \Omega \tag{2.45}$$

The inverse transformation is given by [18]:

$$a = a \tag{2.46}$$

$$e = \sqrt{h^2 + k^2} \tag{2.47}$$

$$i = 2\tan^{-1}\sqrt{p^2 + q^2} \tag{2.48}$$

$$\Omega = \tan^{-1}\left(\frac{p}{q}\right) \tag{2.49}$$

$$\omega = \tan^{-1}\left(\frac{h}{k}\right) - \tan -1\left(\frac{p}{q}\right) \tag{2.50}$$

$$M = \lambda - \tan^{-1}\left(\frac{h}{k}\right) \tag{2.51}$$

$$E = F - \tan^{-1}\left(\frac{h}{k}\right) \tag{2.52}$$

$$\nu = L - \tan^{-1}\left(\frac{h}{k}\right) \tag{2.53}$$

Kepler's equation in terms of equinoctial elements is derived directly from the classical Kepler equation (2.54) using the definitions of λ, F, h, and k [18].

$$M = E - e\sin E \tag{2.54}$$

$$\lambda = F - k\sin F + h\cos F \tag{2.55}$$

The position and velocity vectors in the equinoctial frame can be written by introducing the variables X_1 and Y_1 such that [17, 18]:

$$\boldsymbol{r} = X_1\hat{f} + Y_1\hat{g} \tag{2.56}$$

$$\dot{\boldsymbol{r}} = \dot{X}_1\hat{f} + \dot{Y}_1\hat{g} \tag{2.57}$$

The components of the position and velocity vectors are defined in terms of the equinoctial elements [17, 18]:

$$X_1 = a[(1 - h^2\beta)\cos F + hk\beta \sin F - k] \tag{2.58}$$

$$Y_1 = a[hk\beta \cos F + (1 - k^2\beta)\sin F - h] \tag{2.59}$$

$$\dot{X}_1 = \frac{a^2 n}{r}[hk\beta \cos F - (1 - h^2\beta)\sin F] \tag{2.60}$$

$$\dot{Y}_1 = \frac{a^2 n}{r}[(1 - k^2\beta)\cos F - hk\beta \sin F] \tag{2.61}$$

where $n = \sqrt{\mu/a^3}$ is the mean motion, and $r = a(1 - k\cos F - h\sin F)$ is the orbit radius. Equations 2.58-2.61 introduce the parameter β, which is defined by:

$$\beta = \frac{1}{1 + \sqrt{1 - h^2 - k^2}} \tag{2.62}$$

A state vector containing the six equinoctial elements is defined, $\boldsymbol{z} = [a, h, k, p, q, \lambda]^T$. The rate of change of the equinoctial elements can be written in the same manner that the rate of change of the COEs was written in equation 2.18:

$$\frac{d\boldsymbol{z}}{dt} = \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{r}} \cdot \boldsymbol{v} + \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{v}} \cdot \left(-\frac{\mu}{r^3}\boldsymbol{r} + \boldsymbol{A}\right) = \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{v}} \cdot \boldsymbol{A} \tag{2.63}$$

where $\boldsymbol{A}$ is the acceleration vector expressed in the equinoctial frame. The partial derivatives of the equinoctial elements with respect to velocity are needed in order to obtain the LPEs in terms of the equinoctial elements. The partial derivatives can be obtained through the Poisson brackets (a_α, a_β) of equinoctial elements [18]:

$$\frac{\partial a_a}{\partial \boldsymbol{v}} = -\sum_{\beta=1}^{6}(a_\alpha, a_\beta)\frac{\partial \boldsymbol{r}}{\partial a_\beta} \tag{2.64}$$

where a_α and a_β are equinoctial elements from the state vector $\boldsymbol{z}$. Deriving the Poisson brackets is not an easy task. Broucke and Cefola [16] derived a transformation equation that can be used to find the Poisson brackets of equinoctial elements in terms

of the Poisson brackets of COEs:

$$[(p_\alpha, p_\beta)] = \left(\frac{\partial p_\alpha}{\partial a_\lambda}\right) [(a_\lambda, a_\mu)] \left(\frac{\partial p_\beta}{\partial a_\mu}\right)^T \tag{2.65}$$

where $[(p_\alpha, p_\beta)]$ is the six by six matrix of Poisson brackets in terms of the equinoctial elements, $[(a_\lambda, a_\mu)]$ is the six by six matrix of Poisson brackets in terms of COEs, and the partial derivative terms on the ends are six by six matrices representing the partial derivatives of the equinoctial elements with respect to COEs. With the Poisson brackets at hand, the partials of the equinoctial elements with respect to velocity can be written. This derivation is quite extensive and is included in Appendix A. In the simplest form, however, the LPEs in terms of the equinoctial elements are given by [18]:

$$\frac{da}{dt} = \left(\frac{\partial a}{\partial \dot{\boldsymbol{r}}}\right)^T \cdot \boldsymbol{A} = M_{11}a_f + M_{12}a_g + M_{13}a_w \tag{2.66}$$

$$\frac{dh}{dt} = \left(\frac{\partial h}{\partial \dot{\boldsymbol{r}}}\right)^T \cdot \boldsymbol{A} = M_{21}a_f + M_{22}a_g + M_{23}a_w \tag{2.67}$$

$$\frac{dk}{dt} = \left(\frac{\partial k}{\partial \dot{\boldsymbol{r}}}\right)^T \cdot \boldsymbol{A} = M_{31}a_f + M_{32}a_g + M_{33}a_w \tag{2.68}$$

$$\frac{dp}{dt} = \left(\frac{\partial p}{\partial \dot{\boldsymbol{r}}}\right)^T \cdot \boldsymbol{A} = M_{41}a_f + M_{42}a_g + M_{43}a_w \tag{2.69}$$

$$\frac{dq}{dt} = \left(\frac{\partial q}{\partial \dot{\boldsymbol{r}}}\right)^T \cdot \boldsymbol{A} = M_{51}a_f + M_{52}a_g + M_{53}a_w \tag{2.70}$$

$$\frac{d\lambda}{dt} = n + \left(\frac{\partial \lambda}{\partial \dot{\boldsymbol{r}}}\right)^T \cdot \boldsymbol{A} = n + M_{61}a_f + M_{62}a_g + M_{63}a_w \tag{2.71}$$

where the M_{xx} coefficients are elements of the six by three matrix, M, defined in Appendix A and a_f, a_g, a_w, are the acceleration components in the $\hat{f}$, $\hat{g}$ and $\hat{w}$ directions respectively.

As with all the other sets of equations derived in the previous sections, equations 2.66 - 2.71 could be used to optimize a spacecraft trajectory. The main disadvantage of this set of equations is that it is hard to visualize the orbit in terms of equinoctial

elements. Once a solution is obtained it may be easier to convert back to COEs in order to get a better understanding of the solution. Despite the slight disadvantage, and the burdensome derivation, the equinoctial elements offer a singularity-free set of equations which could be extremely important when using numerical methods.

2.2 Optimal Control Methods

The subject of optimal spacecraft maneuvers has been studied extensively in the past. *Optimal Control Theory* has been applied to a variety of spacecraft trajectory problems in order to solve both minimum-time and minimum-fuel problems. A brief overview of *Optimal Control Theory* and examples of its applications to spacecraft trajectories are provided here.

2.2.1 Optimal Control Theory. The objective of an optimal control problem is to determine a history of control inputs that minimizes a specified cost functional, or performance measure, while satisfying physical constraints applicable to the system. The calculus of variations provides analytical methods of obtaining solutions to optimal control problems, and has been used in the past by legendary scientists and mathematicians such as Newton, Bernoulli, and L'Hopital. [19] The following overview is taken from derivations presented by Kirk [19] and Bryson [20], supplemented by course notes provided by Dr. Jacques [21].

The physical constraints on the system can be written as a set of first-order differential eqiations:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \tag{2.72}$$

where $\boldsymbol{x}$ is the n-dimensional state vector and $\boldsymbol{u}$ is the m-dimensional control vector. The initial conditions of the state vector are usually specified ($\boldsymbol{x}(t_0) = \boldsymbol{x}_0$). Terminal constraints on the state vector are applied using the function:

$$\psi(\boldsymbol{x}(t_f), t_f) = 0 \tag{2.73}$$

The quantity to be minimized, or the cost functional, for a general optimization problem is:

$$J = \phi(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} L(\boldsymbol{x}, \boldsymbol{u}, t)dt \tag{2.74}$$

The cost functional, J, is comprised of two scalar functions. The scalar function ϕ defines the cost associated with the terminal conditions, and is referred to as the *Mayer* cost. The scalar function L defines the cost associated with the values of $\boldsymbol{x}$ and $\boldsymbol{u}$ throughout the trajectory, and is referred to as the *Lagrange* cost. When J contains both a *Mayer* piece and a *Lagrange* piece it is referred to as a *Bolza* cost. The system *Hamiltonian* is defined as:

$$H \equiv L(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^T \boldsymbol{f} \tag{2.75}$$

where $\boldsymbol{\lambda}$ is the n-dimensional vector of Lagrange multipliers, or costates. The optimal solution will be the control history, $\boldsymbol{u}^*$, that produces the optimal trajectory, $\boldsymbol{x}^*$, minimizing the cost functional, J.

In order to include the dynamic constraints and terminal constraints in the cost functional an augmented cost functional is formulated by adjoining equation 2.72 with Lagrange multipliers, $\boldsymbol{\lambda}$, and equation 2.73 with Lagrange multipliers, $\boldsymbol{\nu}$:

$$J_a = \phi(\boldsymbol{x}(t_f), t_f) + \boldsymbol{\nu}^T \psi(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} (L(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^T (\boldsymbol{f} - \dot{\boldsymbol{x}}))dt \tag{2.76}$$

Note that the quantity $(\boldsymbol{f} - \dot{\boldsymbol{x}})$ is identically equal to zero when the dynamics are satisfied, and the quantity ψ is zero when the terminal constraints are satisfied. Therefore, adding these terms does not change the value of the cost functional. To simplify notation a new function, $\Phi(\boldsymbol{x}(t_f), t_f) \equiv \phi(\boldsymbol{x}(t_f), t_f) + \boldsymbol{\nu}^T \psi(\boldsymbol{x}(t_f), t_f)$, is introduced.

A closer look at the integrand in equation 2.76 reveals that there are two separate terms, one of which is the *Hamiltonian*. The other term, $\lambda^T \dot{\boldsymbol{x}}$, can be integrated by

parts. The augmented cost functional then becomes:

$$J_a = \Phi(\boldsymbol{x}(t_f), t_f) - \boldsymbol{\lambda}^T(t_f)\boldsymbol{x}(t_f) + \boldsymbol{\lambda}^T(t_0)\boldsymbol{x}(t_0) + \int_{t_0}^{t_f} (H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) + \dot{\boldsymbol{\lambda}}^T\boldsymbol{x})dt \quad (2.77)$$

The fundamental theorem of the calculus of variations states that in order for $\boldsymbol{x}^*$ to be a minimum or maximum the variation in the cost functional, J_a, must be equal to zero. Taking the variation of J_a without considering variations in the initial conditions ($\delta\boldsymbol{x}(t_0) = 0$) yields:

$$\begin{aligned}\delta J_a =&(\Phi_x(\boldsymbol{x}(t_f), t_f) - \boldsymbol{\lambda}^T(t_f))\delta\boldsymbol{x}(t_f) + (\dot{\Phi}(\boldsymbol{x}(t_f), t_f) + L(\boldsymbol{x}(t_f), \boldsymbol{u}(t_f), t_f))\delta t_f \\ &+ \int_{t_0}^{t_f} ((H_x + \dot{\boldsymbol{\lambda}}^T)\delta\boldsymbol{x} + H_u\delta\boldsymbol{u})dt\end{aligned} \quad (2.78)$$

where the subscripts on the variables denote partial derivatives. In order for δJ_a to be equal to zero all three terms must be equal to zero for any admissible $\delta\boldsymbol{x}$, $\delta\boldsymbol{u}$, and δt_f. Setting the first term in equation 2.78 equal to zero yields a final condition on the Lagrange multipliers, $\boldsymbol{\lambda}$:

$$\boldsymbol{\lambda}^T(t_f) = \frac{\partial\Phi}{\partial x_{t_f}} = \frac{\partial\phi}{\partial x_{t_f}} + \boldsymbol{\nu}^T\frac{\partial\psi}{\partial x_{t_f}} \quad (2.79)$$

The second term yields the transversality condition:

$$\dot{\Phi}(\boldsymbol{x}(t_f), t_f) + L(\boldsymbol{x}(t_f), \boldsymbol{u}(t_f), t_f) = 0 \quad (2.80)$$

In order for the integral term in equation 2.78 to be equal to zero, the integrand itself must be zero. This yields a first order differential equation for the Lagrange multipliers, $\boldsymbol{\lambda}$, and the optimality condition:

$$\dot{\boldsymbol{\lambda}}^T = -\frac{\partial H}{\partial \boldsymbol{x}} \quad (2.81)$$

$$0 = \frac{\partial H}{\partial \boldsymbol{u}} \quad (2.82)$$

Equations 2.81 and 2.82 along with the dynamic constraints, equation 2.72 are known as the Euler-Lagrange (EL) equations, and they represent the first-order necessary conditions for an extremum. The EL equations together with equations 2.73, 2.79, and 2.80 constitute a two-point boundary-value problem (TPBVP) - where initial and terminal conditions on the states are specified, but boundary conditions on the Lagrange multipliers are unknown. For most realistic problems, solutions to the TPBVP can only be found via numerical methods. [12] The main difficulty with solving the TPBVP is that without knowledge of the initial conditions on the Lagrange multipliers equation 2.81 cannot easily be solved. Numerical methods require an initial guess for the solution, which is refined through an iterative process.

2.2.2 Numerical Methods for solving the TPBVP. Several methods exist to solve the TPBVP numerically. The most popular numerical methods seek to discretize the problem, reducing it to a parameter optimization problem, and minimizing the cost functional directly. The method of collocation is one of the most implemented methods of this type. Both the state and control histories are respresented by discrete points on a mesh, and polynomial functions are used to represent histories between mesh points. A collocation point is then picked at the center of each mesh segment, and the slopes of the polynomials at the mesh points and collocation points are constrained such that they match system dynamics. A nonlinear programming (NLP) problem solver is used to enforce all constraints and minimize the cost functional. [12]

Another popular numerical method for solving the TPBVP is via pseudospectral techniques. Although very similar to the collocation method, pseudospectral methods match the state polynomial at a finite number of nodes. Using the values at the nodes, and interpolating between nodes, a polynomial interpolation of the states is constructed, and the state derivatives are computed. The physical state derivatives are also computed through the system dynamics, and the two sets of derivatives are compared. A NLP problem solver is again used to enforce constraints and minimize the cost functional. [12]

2.2.3 Applications of Optimal Control. One of the first published works on optimized spacecraft trajectories was Lawden's *Optimal Trajectories for Space Navigation*, published in 1963 [22]. Lawden solved the problem of optimal impulsive maneuvers using calculus of variation methods. Lawden presented analytical solutions of optimal thrust profiles for rocket trajectories and orbital transfer maneuvers. In his formulation he treated the Lagrange multipliers as components of a "primer vector" whose behavior indicated the optimal direction of an impulsive thrust. Although Lawden did not consider low-thrust systems in solving the optimal control problem, his work is considered the foundation of optimal space trajectories and is a fundamental reference in a vast majority of the literature on this subject. [22]

Building on Lawden's work, Jean-Pierre Marec's book, *Optimal Space Trajectories*, was first published in France in 1973, followed by an English version in 1979 [23]. Marec's book built on Lawden's work by including the consideration of using low-thrust propulsion systems. His book served as a comprehensive compilation of all active research, at that time, regarding spacecraft trajectory optimization. His list of references is endless and, as a result, he included a variety of trajectory optimization problems. He started by providing a parametric optimization example using the popular Hohmann transfer in order to expose the shortcomings of parametric methods. He then discussed the use of functional optimization methods, similar to methods shown in Section 2.2.1, for optimal transfer problems using both impulsive thrust and continuous low thrust. He discussed a variety of problems through functional optimization methods including transfers in both simple and complex gravitational fields and interplanetary rendezvous. Lastly, he introduced the use of celestial mechanics and classical perturbation methods to formulate the optimal control problem in terms of osculating orbital elements. Although Marec's work dates back over 30 years, it still offers a huge variety of applications of optimal control. [23]

Over the years optimal control methods have become more modernized and numerical solutions have become the preferred choice over analytical solutions. However, analytical solutions have been derived for specialized cases. Wiesel and Alfano ad-

dressed the minimum-time transfer between two circular orbits using very low thrust [24, 25]. The main assumption they made that allowed for a closed form solution is that the thrust magnitude is small enough that the orbit's semi-major axis and eccentricity remain fairly constant for a single revolution. The problem was separated into a fast timescale and a slow timescale problem. The fast timescale problem was formulated to determine the small changes in the orbital elements over one revolution, while maximizing the inclination change for a specified semi-major axis change. Solutions from the fast timescale problem were then used to solve the slow timescale problem over multiple revolutions, while taking into account the vehicle's change in mass. [24, 25]

As mentioned in section 2.2.1, one of the most difficult parts of solving an optimal control problem is providing an initial guess for the solution. In his dissertation, Thorne formulated the minimum-time, continuous thrust orbit transfer problem, and used the shooting method to solve the TPBVP [10]. The shooting method involves numerically integrating the equations governing the dynamics of the spacecraft and the Lagrange multipliers, which once again requires a guess for the initial conditions on the Lagrange multipliers. Thorne presented a method for modeling the inital values of the Lagrange multipliers. He first used numerical results from different scenarios to determine the functional relationship between the Lagrange multipliers and two different parameters - the radius of the final orbit and the constant acceleration acting on the spacecraft. Using analytical and empirical methods he derived approximate expressions that define the initial Lagrange multipliers, and analyzed their convergence. He then used these expressions as a reliable means of providing intial conditions on the Lagrange multipliers for the minimum-time continuous thrust orbit transfer. [10]

Equinoctial elements, as derived in section 2.1.3, have also been used in the past for spacecraft trajectory optimization. Jean Kechichian used the non-singular set of elements to solve the optimal low-thrust rendezvous problem using continuous constant acceleration [17]. After formulating the problem, Kechichian used numerical

methods to solve the TPBVP. The problem is posed as a minimum time problem, and solved using a quasi-Newton minimization algorithm.

Rendezvous problems, similar to those in Kechichian's article [17] are, in a way, similar to the responsive maneuvers that are discussed in this thesis. The main difference is that instead of the objective being to rendezvous with another spacecraft, it is to "rendezvous" with a specific location within an orbit. Another example of a rendezvous problem is provided by Hall and Collazo-Perez [26]. They formulated the minimum-time coplanar phasing problem using Euler-Lagrange theory and used a semi-analytic method to solve the TPBVP, similar to the methods presented by Thorne [10]. The phasing maneuver problem solved by Hall and Collazo-Perez is a special case of the rendezvous problem, where the target spacecraft is on the same orbit as the maneuvering spacecraft, and only separated by a phase angle. For the responsive maneuvers discussed in this thesis the target location on the orbit does not necessarily need to be constrained to be on the same orbit as the maneuvering spacecraft. Although much of the literature presented here may not directly relate to the problem posed in this thesis, it provides useful insight in formulating the optimal control problem for responsive maneuvers.

2.3 Feedback Control Methods

Although the methods discussed in the previous section provide the "best" solutions to the trajectory control problems presented in this research, optimal control methods have some disadvantages. The numerical methods used to find the optimal solution can be computationally intensive, and sensitive to user-input initial guesses. Solutions would also need to be recomputed periodically in real time in order to correct for deviations from the optimal trajectory. The objective in using feedback control methods is to find a solution that is close to the optimal solution, but less computationally intensive and easier to implement. The main advantage of feedback control methods is their responsiveness to disturbances in the trajectory that are

not modeled in the dynamics. The following sections provide a brief background on *Lyapunov Theory*, taken from Ilgen [27], and some examples of its applications.

2.3.1 Lyapunov Theory. Lyapunov theory provides a means of developing a control law that can be used to control the trajectory of a maneuvering spacecraft subject to nonlinear dynamics. Ilgen considers a dynamical system of the form [27]:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{2.83}$$

where, as in section 2.2.1, $\boldsymbol{x}$ is the n-dimensional state vector and $\boldsymbol{u}$ is the m-dimensional control vector. The stability of this dynamical system is considered about an equilibrium point, $\boldsymbol{x}^*$. The vector function, $\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{x}^*$ is defined. Lyapunov's theorem states that there exists a scalar function, $V(\boldsymbol{z})$, with continuous first derivatives, that meets the following characteristics [27]:

1. $V(\boldsymbol{z})$ is positive definite
2. $\dot{V}(\boldsymbol{z})$ is negative definite
3. $V(\boldsymbol{z}) \to \infty$ as $|\boldsymbol{z}| \to \infty$

The function V is then called a Lyapunov function and $\boldsymbol{x}^*$ is globally asymptotically stable, guaranteeing that the equilibrium point will eventually be reached. Ilgen continues by considering a Lyapunov optimal feedback control law, $\boldsymbol{u}^*$, that makes $\dot{V}$ as negative as possible, by minimizing the Lie derivative, $L_f V$, therefore reaching the equilibrium point as fast as possible. The Lie derivative is defined by [27]:

$$L_f V = \left(\frac{\partial V}{\partial \boldsymbol{x}} \right) \boldsymbol{f} \tag{2.84}$$

Ilgen points out that the Lyapunov optimal control law is not necessarily equivalent to the true optimal control law that is computed through solving the TPBVP as described in section 2.2.2. However, it is possible to derive a Lyapunov optimal control law that is very close, if not equivalent, to the optimal control law.

2.3.2 Applications of Feedback Control. Ilgen applied the theory presented in his paper to derive a guidance law for low-thrust orbit transfers between orbits of different semi-major axis, eccentricity, and inclination. He presented formulations of the guidance law in terms of both classical orbital elements and equinoctial elements, using a Lyapunov function that measures the difference between the osculating orbital elements and the target orbital elements. The guidance law determined the thrust direction necessary to minimize the Lie derivative at every time step along the trajectory. He then discussed the optimization of the guidance law by adjusting a number of "free parameters", which are essentially gains in the Lyapunov function. He demonstrated the use of the guidance law by analyzing two scenarios, and compared them to optimal results. [27]

Feedback control methods, such as those presented by Ilgen, have been studied by several authors. In his thesis, Naasz studied the use of feedback control laws for use in spacecraft maneuvers and formation flying. He presented formulations of a Lyapunov-based feedback control law in terms of cartesian coordinates, classical orbital elements, and equinoctial elements, using Lyapunov functions similar to those used by Ilgen. He also presented a way of controlling the position of the spacecraft within the orbit through mean motion control. He applied the derived control laws to demonstrate orbit raising maneuvers, inclination change maneuvers, phasing maneuvers, and finally formation establishment and formation keeping maneuvers. [28]

Gurfil also used a Lyapunov function to derive a nonlinear feedback controller for low-thrust orbital maneuvers. Although the control law derived was not different from the control laws used by Ilgen and Naasz, Gurfil included a mathematical discussion on the controllability of the equations of motion in terms of the classical orbital elements. Gurfil's discussion proves the asymptotic stability of the equilibrium point, as stated by Ilgen. Gurfil demonstrated the performance of the low-thrust nonlinear control law by performing an orbital transfer between geosynchronous orbits, and compared the results to an impulsive maneuver. He concluded that the low-thrust control law was more fuel efficient than an impulsive maneuver for the same orbit transfer. [29]

Another popular article on the topic of nonlinear feedback control is by Petropoulos, who used a Lyapunov function termed the "proximity quotient", or Q-law. As the Lyapunov functions used by Ilgen and Naasz, his Lyapunov function quantifies the proximity of the osculating orbit to the target orbit. Petropoulos, however, included a method in his algorithm that would allow the spacecraft to coast if the effectivity, defined by the rate of change of the Q-law, of the thrust is below some threshold value. He applied his algorithm to demonstrate four different example orbit transfers, showing a wide range of capability. [30]

One of the subjects that Ilgen mentioned was the relationship between the Lyapunov control law and the true optimal control law. In his article, Yang presents both control laws and it can be seen that the Lagrange multipliers, or costates, in the optimal control law are related to the partial derivatives of the Lyapunov function in the Lyapunov control law. Yang demonstrated that using the Lyapunov controller to produce an initial guess for the optimization problem improved the convergence and robustness of the numerical methods he used to solve for the optimal control law. [31] Yang's conclusions validate Ilgen's statement that the Lyapunov control law is very close, if not equivalent, to the optimal control law.

Other control methods have also been used for maneuvers that have similar objectives to those discussed in this thesis. Jean and de Lafontaine presented guidance algorithms, in cubic spline and quatric form, for an autonomous Earth observation satellite using low thrust [32]. The cubic spline control law presented in their paper was initally presented by Guelman and Kogan, who also showed that its performance was optimal [33]. Both papers discussed maneuvers that placed a satellite over a particular terrestrial target, but their methods were specific to the problems being examined. Guelman and Kogan investigated the use of low thrust propulsion to maneuver a spacecraft such that it overflew a series of ground targets over a specified time period [33]. They showed that the applied cubic spline control law produced a fuel-optimal solution, and that the application of electric propulsion was practical [33]. Jean and de Lafontaine applied the same cubic spline law to an Earth observation

satellite, with the objective of overflying a given terrestrial target while maintaining a sun-synchronous orbit [32]. They extended Guelman and Kogan's research by accounting for perturbations due to air drag and Earth oblateness, and by investigating the use of a quartic guidance law [32]. The methods discussed in these two papers could be applied to the algorithms developed in this research in order to solve the full target overflight problem.

Overall, feedback control methods could have a significant advantage over optimal control methods. Since feedback control methods can produce optimal results, the simplicity of implementing feedback control makes them more appealing than optimal control methods. The literature presented above shows that feedback control methods have a broad range of applications, and, as such, provide flexibility in capabilities that could be implemented. As will be discussed in Chapters III and IV, both algorithms are developed, and the feedback control algorithm tends to be more user-friendly.

2.4 Summary

Chapter II presented the necessary background information in order to fully understand the problem posed in this research. The equations of motion for a low-thrust maneuvering spacecraft were presented in cartesian coordinates, classical orbital elements, and equinoctial elements. The optimal control methods and feedback control methods that are being considered in this research were briefly described, and examples of their applications were presented. The following chapter discusses how the above methods can be applied to the specific problem at hand.

III. Model Development and Methodology

The following sections present the methodology in setting up models for computing maneuver trajectories using optimal control methods and feedback control methods. Chapter II presented a variety of variables that can be used to define the equations of motion (EOM). The classical orbital elements (COEs) were selected for these particular models due to their slowly-changing behavior, and intuitiveness in orbit visualization. For the maneuvers discussed throughout this research, the spacecraft begins at a circular, or near-circular, orbit and all of the intial COEs are known. The code associated with these algorithms is provided in Appendices B - D.

3.1 Optimal Control Formulation

The minimum-time transfer problem for a low-thrust maneuvering spacecraft is formulated using Euler-Lagrange theory as described in section 2.2.1. The first-order differential equations defining the physical constraints on the system are the EOM in terms of the COEs as shown in equations 2.28-2.33. As was discussed in Chapter II these equations are singular for orbits of zero eccentricity. The singularity immediately causes a problem since the spacecraft starts on a circular orbit, and a work-around must be implemented before continuing. The assumption can be made that the orbit remains circular throughout the trajectory. Since the amount of thrust considered for the maneuvers is very small, it is expected that the change in eccentricity will be very small. Therefore, the assumption that the orbit remains fairly circular is safe. However, in order to fully declare the validity of the assumption the solution must be checked to verify that the eccentricity remains close to zero throughout the trajectory.

Setting $e = 0$ simplifies some of the terms in equations 2.28 and 2.29, and since the eccentricity throughout the trajectory is assumed to be zero, equation 2.29 can be ignored completely. Equations 2.32 and 2.33, that have e in the denominator, define the rate of change for the argument of perigee and true anomaly, respectively. The argument of perigee(ω) is measured from the orbit's eccentricity vector and, since the orbit's eccentricity is zero, it is undefined. As a result, equation 2.32 can

be ignored. The true anomaly(ν) is measured from perigee and is therefore also undefined. The position of a spacecraft within a circular orbit is measured by the argument of latitude, which is referred to as ν throughout this thesis, and its rate of change can be approximated by the mean motion of the osculating orbit. The EOM then simplifies to:

$$\frac{da}{dt} = \frac{2}{n} a_\theta \tag{3.1}$$

$$\frac{di}{dt} = \frac{\cos\nu}{na} a_h \tag{3.2}$$

$$\frac{d\Omega}{dt} = \frac{\sin\nu}{na\sin i} a_h \tag{3.3}$$

$$\frac{d\nu}{dt} = n \tag{3.4}$$

where $n = \sqrt{\mu/a^3}$ is the mean motion of the osculating orbit.

The acceleration vector for this set of equations is defined in the Local-Vertical-Local Horizontal (LVLH) frame as shown in equation 2.19. The orientation of the acceleration can be defined by two control angles, θ and β. As shown in Figure 3.1, the angle θ is the in-plane angle and is measured from the $\hat{i}_r$ unit vector, and β is the out-of-plane angle. The acceleration components (a_r, a_θ, a_h) can then be written as functions of the constant acceleration magnitude and the control angles θ and β:

$$a_r = A\cos\beta\cos\theta \tag{3.5}$$

$$a_\theta = A\cos\beta\sin\theta \tag{3.6}$$

$$a_h = A\sin\beta \tag{3.7}$$

Equations 3.5-3.7 can then be substituted into the EOM so that the physical constraints are functions of the classical orbital elements and the control angles.

The problem can be further simplified by only considering coplanar maneuvers ($\beta = 0$). Equations 3.2 and 3.3 then vanish and the EOM only consists of one equation

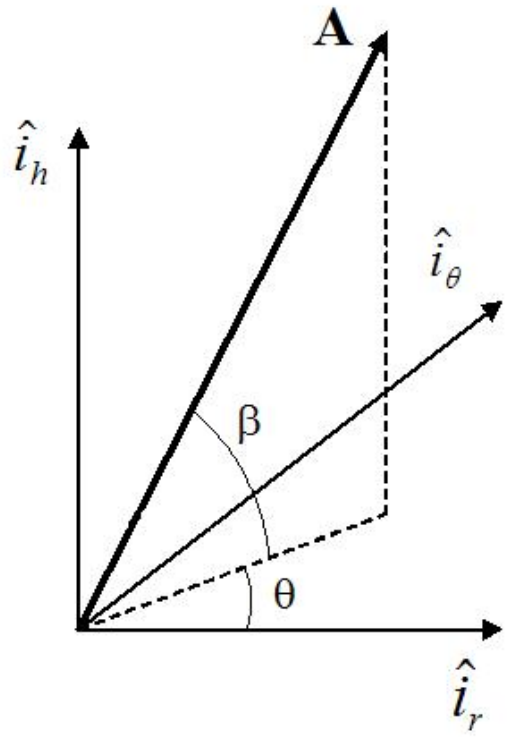

Figure 3.1: Acceleration vector in LVLH frame

for a and one equation for ν:

$$\frac{da}{dt} = \frac{2}{\sqrt{\mu}} a^{3/2} A \sin\theta \tag{3.8}$$

$$\frac{d\nu}{dt} = \sqrt{\frac{\mu}{a^3}} \tag{3.9}$$

The initial conditions on the states, a_0 and ν_0, are known. The final conditions for semi-major axis, a_f, could be defined if the final altitude of the spacecraft needs to be constrained. The final condition for the argument of latitude, ν_f, is defined and is the final position within the orbit that the spacecraft is required to reach.

3.1.1 The Cost Functional. The simplest form of the cost functional for the minimum-time transfer problem is the Mayer form, which can be written as:

$$J = t_f \tag{3.10}$$

However, since numerical methods will be used to find a solution, a Lagrange cost term should be added to the cost functional. The purpose of the additional term is to avoid discontinuities in the optimal control solution that could result in convergence issues for the nonlinear programming (NLP) solver. The cost functional then becomes:

$$J = t_f + \int_{t_0}^{t_f} \alpha\theta^2 dt \tag{3.11}$$

where θ is the control angle, and α is a weight factor. The added Lagrange cost term actually penalizes control usage and if it is allowed to be too large the spacecraft will exhibit very slow changes in the control angle, which would be undesirable. Therefore, the value of α should be set as close to zero as possible, while still allowing the NLP solver to converge to a solution. With α being very small the final Lagrange cost will be negligible and the cost functional is mostly dependent on the value of the final time(t_f), which is being minimized.

3.1.2 The Euler-Lagrange Equations. Based on the cost functional in equation 3.11 the Hamiltonian can be written using equation 2.75:

$$H = \alpha\theta^2 + \lambda_1 \left(\frac{2}{\sqrt{\mu}} a^{3/2} A \sin\theta \right) + \lambda_2 \left(\sqrt{\frac{\mu}{a^3}} \right) \tag{3.12}$$

The Euler-Lagrange(EL) equations can then be written using equations 2.81 and 2.82:

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial a} = -\lambda_1 \left(\frac{3}{\sqrt{\mu}} a^{1/2} A \sin\theta \right) + \lambda_2 \left(\frac{3}{2} \sqrt{\frac{\mu}{a^5}} \right) \tag{3.13}$$

$$\dot{\lambda}_2 = -\frac{\partial H}{\partial \nu} = 0 \tag{3.14}$$

$$0 = \frac{\partial H}{\partial \theta} = 2\alpha\theta + \lambda_1 \left(\frac{2}{\sqrt{\mu}} a^{3/2} A \cos\theta \right) \tag{3.15}$$

From the EL equations it can be easily seen that the second Lagrange multiplier, λ_2, is constant. Without any boundary conditions on λ_1, however, the equations cannot be integrated to obtain a solution. Therefore, the problem is solved numerically using pseudospectral methods.

3.1.3 Algorithm Setup. The problem is fomulated as a single phase problem. The algorithm includes a sparse nonlinear optimal (SNOPT) solver that is used to solve the problem directly [34]. Given an initial guess for the control and the states, the algorithm discretizes the problem using gauss pseudospectral methods and verifies optimality conditions at each iteration step until a solution is obtained within desired tolerances. The most challenging part is generating an initial guess. To obtain a refined optimal control solution the algorithm runs twice for this problem. The first time a very coarse grid is used with only five grid refinement iterations and relatively low tolerances of 1e-2. The solution from this optimization is then used as an initial guess for the second run. The second optimization run includes a much finer grid with 25 grid refinement iterations and higher tolerances of 1e-3.

The optimal control solution is then used to numerically integrate the EOM (equations 2.28-2.33), and obtain the transfer trajectory. However, the time vector that is output from the optimization algorithm is not the same size as the time vector used in the numerical integration. In order to be able to pick the correct value of the optimal control angle throughout the trajectory, a cubic spline is fit to the optimal control curve and evaluated at the appropriate time value throughout the integration. The resulting transfer trajectory is compared to a reference trajectory of a non-maneuvering spacecraft at the same initial conditions. The time difference, Δt, is defined as the difference between the time it took the maneuvering spacecraft to reach the final position (ν_f) versus the time it would have taken the reference spacecraft to reach the same position, and is calculated by:

$$\Delta t = \nu_f \sqrt{\frac{a_0^3}{\mu}} - t_f \tag{3.16}$$

where ν_f is the desired final position, and t_f is the duration of the maneuver, or the Mayer cost of the optimal solution. The eccentricity of the transfer trajectory can also be plotted to validate the earlier assumption that the osculating orbit remains fairly circular.

3.2 Feedback Control Formulation

Lyapunov theory is applied to the nonlinear EOM (equations 2.28-2.33) in order to control the trajectory of the maneuvering spacecraft, as shown by Ilgen [27] and Naasz [28]. First the EOM is written in matrix form:

$$\dot{\boldsymbol{x}} = N\boldsymbol{u} \tag{3.17}$$

where $\boldsymbol{u} = [a_r, a_\theta, a_h]^T$ and N is the six by three matrix comprised of the coefficients in equations 2.28-2.33:

$$N = \begin{bmatrix} \frac{2a^2 e \sin\nu}{h} & \frac{2a^2 p}{hr} & 0 \\ \frac{p\sin\nu}{h} & \frac{(p+r)\cos\nu + re}{h} & 0 \\ 0 & 0 & \frac{r\cos(\omega+\nu)}{h} \\ 0 & 0 & \frac{r\sin(\omega+\nu)}{h\sin i} \\ -\frac{p\cos\nu}{he} & \frac{(p+r)\sin\nu}{he} & -\frac{r\sin(\omega+\nu)\cos i}{h\sin i} \\ \frac{p\cos\nu}{he} & -\frac{(p+r)\sin\nu}{he} & 0 \end{bmatrix} \tag{3.18}$$

The equation for ν, however, includes an additional term that is equivalent to the mean motion of a circular orbit. The EOM is then written as:

$$\dot{\boldsymbol{x}} = N\boldsymbol{u} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{h}{r^2} \end{bmatrix} \tag{3.19}$$

The Lyapunov function, V, is based on the difference between the osculating COEs and the desired COEs and is defined as:

$$\begin{aligned} V =& \frac{1}{2}\left(K_1\frac{(a-a^*)^2}{R_e^2} + K_2(e-e^*)^2 + K_3(i-i^*)^2 + K_4(\Omega-\Omega^*)^2 + \right. \\ & \left. K_5(\omega-\omega^*)^2 + K_6(\nu-\nu^*)^2\right) \end{aligned} \tag{3.20}$$

where the starred values define the COEs of the desired final orbit, and $K_{()}$ represents a positive gain associated with the COE being controlled. Since V is a function of the squared difference between the osculating COEs and the desired COEs, it is positive definite. The partial of the Lyapunov function with respect to the COEs used in the Lie derivative is then written as:

$$\frac{\partial V}{\partial \boldsymbol{x}} = V_x = \begin{bmatrix} K_1\frac{(a-a^*)}{R_e^2} & K_2(e-e^*) & K_3(i-i^*) & K_4(\Omega-\Omega^*) & K_5(\omega-\omega^*) & K_6(\nu-\nu^*) \end{bmatrix} \tag{3.21}$$

The control law is then defined as:

$$\boldsymbol{u} = -A\frac{N^T V_x^T}{|N^T V_x^T|} \tag{3.22}$$

where A is the constant acceleration magnitude, and the fraction term is a control unit vector defining the direction of the acceleration. The EOM can then be numerically integrated and the control acceleration vector is calculated at every iteration. As was done in the optimal control formulation, the rate of change of ω can be set to zero, and the rate of change of ν can be approximated by the mean motion to avoid singularities in the EOM.

3.2.1 Gain Selection. Although the control law presented in equation 3.22 guarantees a globally asymptotically stable trajectory, its performance is somewhat dependent on the values selected for the gains, $K_{()}$. Intuitively, the gain value is related to how quickly the target value for the respective COE is reached. For example, if a large gain value was selected for K_3 the controller's focus would be changing the

inclination. On the other hand, if a COE is to be left uncontrolled, its respective gain should be set to zero. Ilgen performed a short sensitivity analysis that showed that the performance of the guidance law is mildly sensitive to the gain values [27]. He showed that for the maneuvers he analyzed, changing the gain values would only change the maneuver duration by a few hours, which is a small change since the maneuvers lasted 192 days. Since some small amount of sensitivity is present, however, it is conceivable that there exists a value for each $K_{()}$ that makes the Lyapunov control law equivalent to the true optimal control law. Naasz briefly discusses using time-varying gains that maximize the change in each COE [28]. His derivation is specific to his problem, but the concept he presents could be useful in future work. The topic of "optimal" gain selection is not discussed further in this thesis, but is one of the recommendations for future study.

3.2.2 Controlling the Position of the Spacecraft Within the Orbit. Due to the mean motion term in equation 3.19, it is not possible to control ν directly. Naasz shows a way that the position of the spacecraft within the orbit can be controlled through the mean motion [28]. He states that for an uncontrolled spacecraft in a circular orbit:

$$\dot{\nu} = n \tag{3.23}$$

and the relative dynamics are:

$$\frac{d}{dt}(\nu - \nu^*) = n - n^* = \sqrt{\frac{\mu}{a^3}} - \sqrt{\frac{\mu}{a^{*3}}} = -K_\nu(\nu - \nu^*) \tag{3.24}$$

where K_ν is some positive gain. Solving for a results in a new value for the desired semi-major axis, that will also correct the spacecraft's position within the orbit:

$$a^{**} = \left(-\frac{K_\nu}{\sqrt{\mu}}(\nu - \nu^*) + \frac{1}{a^{*3/2}}\right)^{-2/3} \tag{3.25}$$

Naasz points out that as $\nu - \nu^*$ approaches zero the value of the desired semi-major axis approaches the original value (a^*). His simulation results show that the above method

of mean motion control should only be used after the desired orbit plane orientation is achieved. If mean motion control is applied from the start of the maneuver the controller's focus would be fixing the mean motion, and the rate of change of other COEs would be small. Naasz therefore suggests that the orbit plane orientation should be corrected first, and then mean motion control can be applied to correct the spacecraft's phase. [28]

The feedback control algorithm described above provides the ability to command values for every orbital element, and can be used for both coplanar and out-of-plane maneuvers. The output trajectory can be compared to a reference trajectory and the Δt can be calculated. Due to the simplicity of the algorithm the implementation of further capabilities is fairly straight-forward, whereas implementing such capabilitites in an optimization algorithm may not be as intuitive. The addition of a thrust-coast duty cycle is worth considering because it would provide flexibility in the applications of this algorithm.

3.2.3 Thrust-coast Duty Cycle. The implementation of a thrust-coast duty cycle applies to spacecraft that are not equipped with a propulsion system that has the ability to thrust continuously. The duty cycle is based on timing - the spacecraft is allowed to thrust for a given pulse duration and then must drift for a certain amount of time before pulsing again. At each iteration of the numerical integration, the algorithm checks the history of the control input and decides whether to thrust or not based on the given duty cycle. The Lyapunov controller still guarantees that the spacecraft will reach the desired final state, but it will take a longer amount of time to do so.

3.3 Defining Desired Final Spacecraft Position

Sections 3.1 and 3.2 presented the details in setting up the algorithms that are used throughout this research. One topic that was not discussed in the previous sections, however, is how the final position of the spacecraft within the orbit (ν_f) is

defined. The angle ν_f should be selected such that when the spacecraft reaches it, the desired ground target is overflown. Given the coordinates of the desired ground target a position unit vector in the Earth-Centered-Fixed (ECF) frame can be written as:

$$x_{ECF} = \cos\lambda_{tgt}\cos\phi_{tgt} \tag{3.26}$$

$$y_{ECF} = \cos\lambda_{tgt}\cos\phi_{tgt} \tag{3.27}$$

$$z_{ECF} = \sin\lambda_{tgt} \tag{3.28}$$

where λ_{tgt} is the target latitude and ϕ_{tgt} is the target longitude. The position vector can then be transformed into the Earth-Centered-Inertial (ECI) frame through a single 3-axis rotation through the Greenwich sidereal time, γ, at the specified time of overflight. The angle γ can be calculated using the known Greenwich sidereal time at epoch and adding the angle through which the Earth has rotated by the time of overflight:

$$\gamma = \gamma_g + \omega_e(t_{of} - t_{ep}) \tag{3.29}$$

where t_{of} is the specified overflight time, t_{ep} is the epoch time, γ_g is the Greenwich sidereal time at epoch, and ω_e is the Earth's rotation rate. The postition vector can then be transformed into the ECI frame by:

$$\boldsymbol{r}_{ECI} = R^{IF}\begin{bmatrix} x_{ECF} \\ y_{ECF} \\ z_{ECF} \end{bmatrix} \tag{3.30}$$

where R^{IF} is the 3-axis rotation matrix defined as:

$$R^{IF} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.31}$$

The position vector can finally be transformed into the perifocal frame using a 3-1-3 rotation sequence through the angles ω, i, and Ω respectively, defining the orientation of the final orbit plane. For coplanar maneuvers the orbital plane does not change orientation and these values are the same as the initial conditions. If out-of-plane maneuvers are considered, these values would have to be calculated. The position vector in the perifocal frame can be written as:

$$\boldsymbol{r}_p = R^{PI}\boldsymbol{r}_{ECI} \tag{3.32}$$

where R^{PI} is the 3-1-3 rotation matrix defined as [11]:

$$R^{PI} = \begin{bmatrix} \cos\omega\cos\Omega - \sin\omega\sin\Omega\cos i & \cos\omega\sin\Omega + \sin\omega\cos\Omega\cos i & \sin\omega\sin i \\ -\sin\omega\cos\Omega - \cos\omega\sin\Omega\cos i & -\sin\omega\sin\Omega + \cos\omega\cos\Omega\cos i & \cos\omega\sin i \\ \sin\Omega\sin i & -\cos\Omega\sin i & \cos i \end{bmatrix} \tag{3.33}$$

For cases where the orbit plane orientation needs to be calculated, Escobal provides equations that can be used to calculate the appropriate i and Ω such that the spacecraft will overfly the ground target at the specified time. He also shows that the value of ω can be arbitrarily set to zero without changing the solution. [35]

With the position vector written in the perifocal frame the value of ν_f can be calculated through general trigonometry:

$$\nu_f = \tan^{-1}\frac{y_p}{x_p} \tag{3.34}$$

where x_p and y_p are the first and second components of the perifocal position vector. It is important to note that the method described above assumes that at the specified overflight time the ground target is on the plane of the osculating orbit. In order to fully solve the target overflight problem the time at which the target crosses the plane of the orbit would need to be calculated, which is not in the scope of this thesis. The target overflight problem has been discussed by other authors in the past, and

methods have been presented that could, in the future, be applied here [32, 33]. The main focus of this research is the Δt created by the maneuvering spacecraft, and the value of ν_f is selected arbitrarily.

3.4 Summary

Chapter III presented the details in setting up the two models used throughout this research. The minimum-time optimal control problem for the simplified EOM was formulated using Euler-Lagrange theory, and the set-up of the pseudospectral algorithm used to obtain the solution was discussed. For the feedback control model, a control law was presented using a Lyapunov function that quantifies the proximity of the osculating orbit to the target orbit. The next chapter presents the results of multiple simulations that were run using the algorithms described, including the scenario simulation that makes up the third phase of this research.

IV. Results and Discussion

This chapter presents the results of several test cases run using the algorithms developed as described in Chapter III. First, in-plane maneuvers using optimal control are presented. A simple in-plane maneuver using feedback control is presented, and compared to the optimal maneuver. Out-of-plane maneuverability, and the thrust-coast duty cycle implementation through feedback control are also demonstrated. Using the intuitive results for in-plane maneuvers, an analytical method is employed to derive an equation for the time of arrival difference, Δt, created by a maneuvering spacecraft. Finally using the analytical result and feedback control methods a spacecraft maneuver scenario is presented.

4.1 Optimal Control Results

The optimal control algorithm for in-plane maneuvers is used to run four test cases. The first three cases are phasing maneuvers, where the spacecraft's objective is to reach ν_f while its final altitude is constrained to be equal to the initial altitude. Phasing maneuvers could be useful if the spacecraft's altitude at the target point is constrained for some reason but, as the following results show, they do not provide the maximum amount of Δt achievable. The final test case removes the final altitude constraint and allows the spacecraft to lower its altitude to no less than 200 km until ν_f is reached. The minimum altitude at 200 km is chosen to avoid causing a trajectory that intersects the Earth's surface. All maneuvers use a constant thrust of $1N$ for a spacecraft with initial mass of $1000kg$. Assuming that the change in mass throughout the maneuver is negligible, the corresponding constant acceleration is $10^{-6}km/s^2$.

4.1.1 Test Case 1. For the first case, the spacecraft starts on a circular orbit with altitude of $1000km$, which is equivalent to $a = 7378km$, inclined at $i = 45^o$, and $\Omega = \omega = 0^o$. The target position is selected within the spacecraft's first orbital period at $\nu_f = 180^o$. As explained in Section 3.1, the control weight factor, α, should be set as close to zero as possible while still allowing the nonlinear programming (NLP) solver to converge to a solution. For the phasing maneuver being analyzed, it is expected

that the control angle, θ, must change direction half way through the maneuver in order to return the spacecraft to its original altitude. Intuitively, the spacecraft should first thrust against the velocity direction ($\theta = -90^o$) to lower its orbit and switch to thrusting along the velocity direction ($\theta = 90^o$) to raise its orbit back to the original altitude. Therefore, a jump discontinuity is expected in the optimal control solution. With $\alpha = 0$, the NLP has difficulties converging due to the jump discontinuity. Figure 4.1 depicts the dependence of the optimal control solution on α. The curve

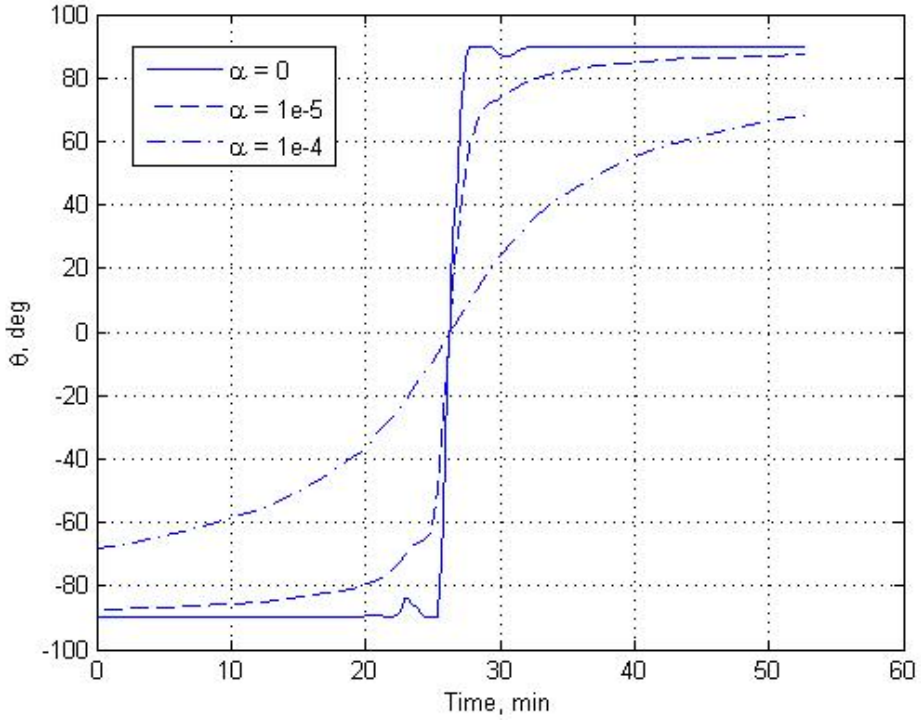

Figure 4.1: Control curve behavior for values of α

corresponding to $\alpha = 0$ is the "true" optimal solution because it does not include an additional Lagrange term in the cost functional being minimized. Therefore, the total cost is only dependent on the duration of the maneuver. However, the curve exhibits some small spikes near the jump discontinuity, creating difficulties for the NLP solver. The plot was created by forcing the algorithm to stop after a small number of grid refinement iterations so that the solution could be analyzed. When the NLP solver was allowed to run for the full 25 iterations it could not converge to a solution. The

difficulties with the jump discontinuity were even more apparent in cases where the maneuvers lasted for longer periods of time. The curve corresponding to $\alpha = 10^{-5}$ seems smoother, very close to the "true" optimal curve, and allows the NLP solver to converge to a solution. The Mayer cost associated with this solution was equivalent to the "true" optimal case, but this solution included a very small Lagrange cost. The final curve, corresponding to $\alpha = 10^{-4}$, is even smoother but exhibits an unrealistic quality. The optimal control angle is shown to start at about -70^o and end at 70^o in just over 50 minutes. Although 50 minutes is not a long amount of time in orbit, a realistic spacecraft would surely have the capability to change its thrust direction much faster. In this case, the small rate of change for the control angle enforced by α resulted in a Mayer cost that was slightly higher than the "true" optimal case. As a result, $\alpha = 10^{-5}$ is chosen as the appropriate value for this case.

Figure 4.2 shows the optimal solution for this scenario. The top left plot shows

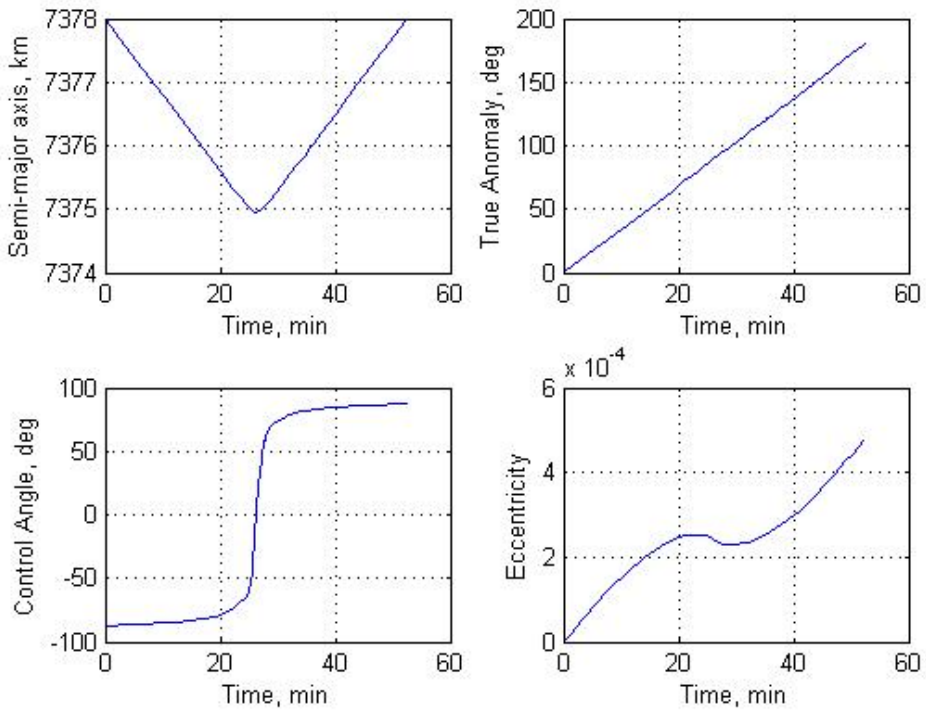

Figure 4.2: Optimal Solution for Test Case 1

that the spacecraft's semi-major axis is lowered by slightly over $3km$ throughout the maneuver, and the bottom left plot shows that the control angle starts at -90^o and ends at 90^o, as expected. The bottom right plot shows that the change in eccentricity throughout the maneuver is on the order of 10^{-4}, which validates the assumption made in Chapter III that the orbit remains fairly circular. The duration of the maneuver is given by the Mayer cost, which is $3152.5sec$ or $52.5min$ for this case. The additional Lagrange cost enforced by the weight factor, α, is 0.0631 and, as expected, it is negligible compared to the Mayer cost. The corresponding ΔV can then be calculated by multiplying the duration of the maneuver by the constant acceleration magnitude, and for this case it is $0.0032km/s$. Figure 4.3 shows the ground track of the reference orbit and depicts the position of the maneuvering spacecraft at the end of the maneuver, and the corresponding position of the spacecraft on the reference orbit. Plotting the final position of the maneuvering spacecraft versus the reference

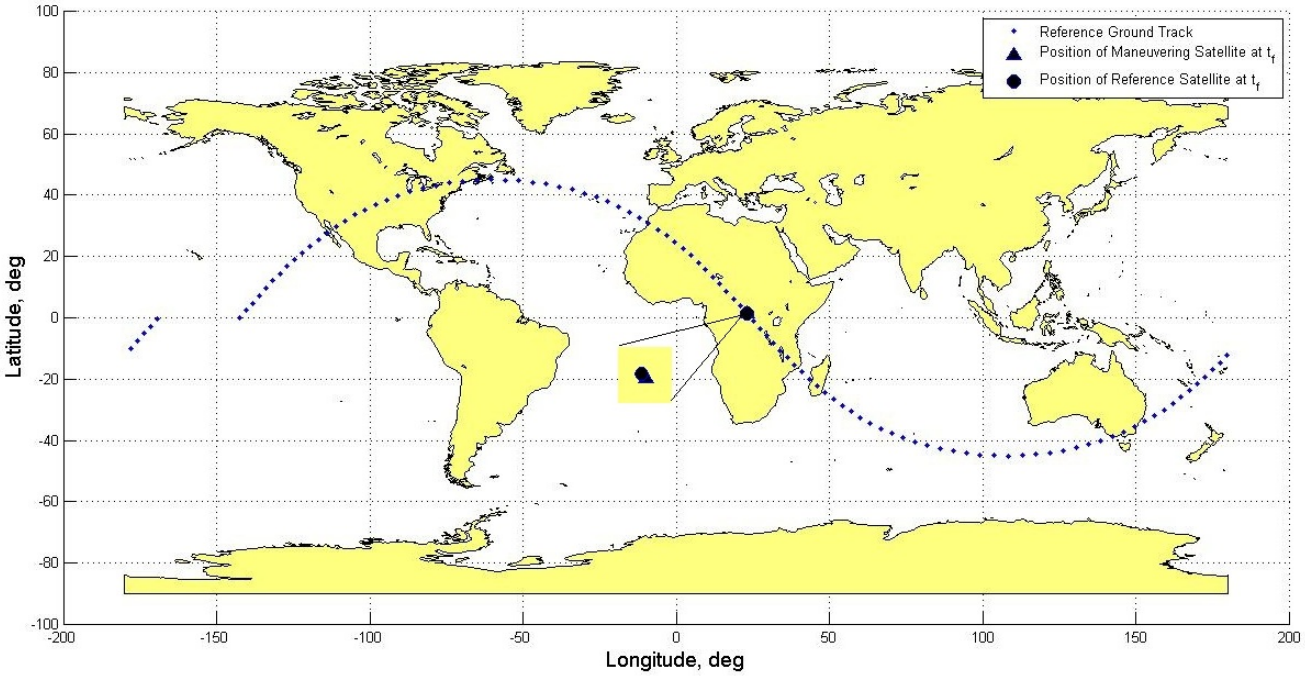

Figure 4.3: Ground Track for Test Case 1

spacecraft provides a visual depiction of the Δt that is used to quantify the effect of the maneuver. As can be seen in Figure 4.3, after maneuvering for almost one hour

the position markers are on top of each other, and the time difference created is not visible. The Δt for this case, calculated using equation 3.16, is $0.97sec$. Therefore, the spacecraft must be allowed to thrust for a longer period of time to create a larger Δt.

4.1.2 Test Case 2. The second case considers a spacecraft in circular orbit at an altitude of $500km$, or $a = 6878km$, inclined at 65^o, and $\Omega = \omega = 0^o$. In order to allow the spacecraft to thrust for a longer period of time the target position is chosen at 180^o within the spacecraft's third orbital period, or $\nu_f = 900^o$. The appropriate value of the control weight factor is selected using the same technique described in Section 4.1.1, and is determined to be $\alpha = 10^{-5}$. Figure 4.4 shows the optimal solution for this scenario. The bottom left plot, the control angle throughout the maneuver,

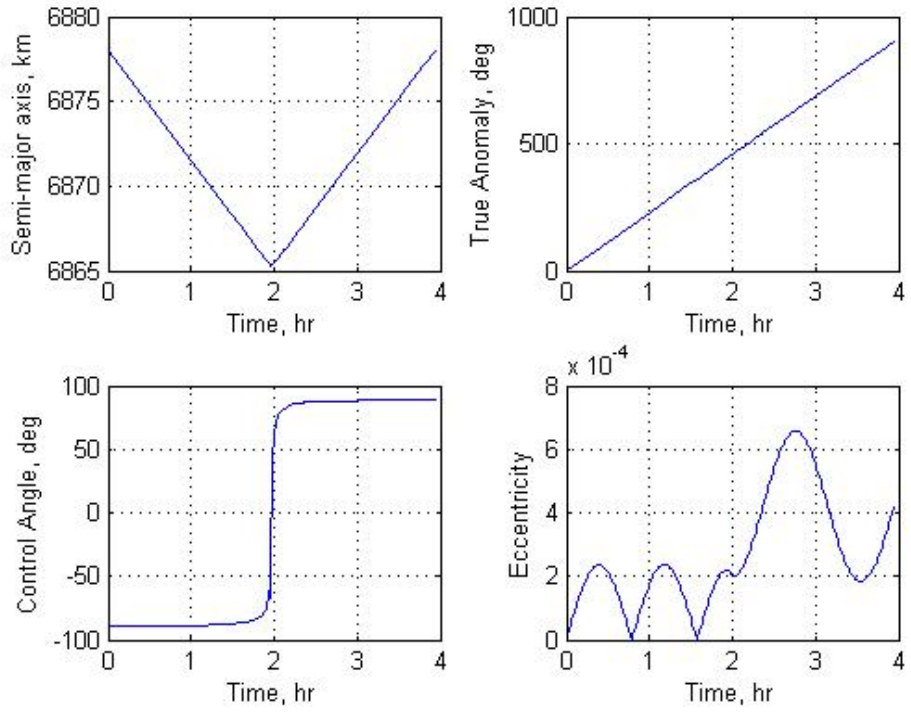

Figure 4.4: Optimal Solution for Test Case 2

shows the same behavior as was seen in the first case, and the spacecraft's altitude was lowered by about $13km$ as can be seen from the top left plot. The eccentricity of the

orbit again remains very close to zero, as seen in the bottom right plot. The duration of the maneuver for this case is $14,172sec$, or $3.9hr$, and the corresponding ΔV is $0.014km/s$. The additional Lagrange cost is 0.3274, which once again is negligible compared to the Mayer cost. Figure 4.5 shows the final orbit ground track of the reference spacecraft and depicts the position of the maneuvering spacecraft at the end of the maneuver, and the corresponding position of the spacecraft on the reference orbit. The position markers in Figure 4.5 show a very small separation between the

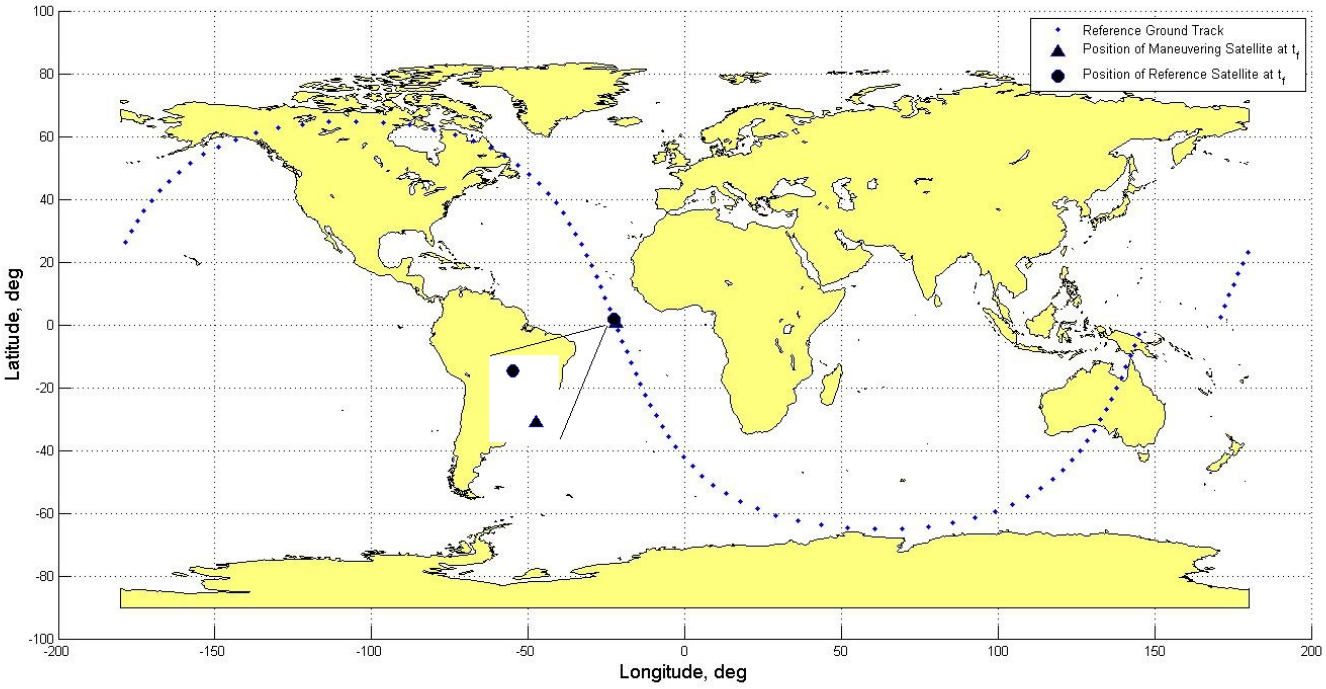

Figure 4.5: Ground Track for Test Case 2

maneuvering and reference spacecraft after maneuvering for almost four hours. The Δt for this case is $20sec$, which could be a significant change based on the mission requirements. However, if the requirement is to create a Δt in the order of minutes, or hours, the spacecraft must be allowed to thrust for a significantly longer amount of time.

4.1.3 Test Case 3. The third case considers a spacecraft in circular polar orbit ($i = 90^o$) at an altitude of $500km$, or $a = 6878km$, with $\Omega = \omega = 0^o$. The target position is chosen at 180^o within the spacecraft's 15^{th} orbital period, or $\nu_f = 5220^o$, in order to allow the spacecraft to thrust for a significant amount of time. The appropriate value of the control weight factor is selected using the same technique described in Section 4.1.1, and is determined to be $\alpha = 10^{-4}$. Figure 4.6 shows the optimal solution for this scenario. As in the previous two cases, the control angle

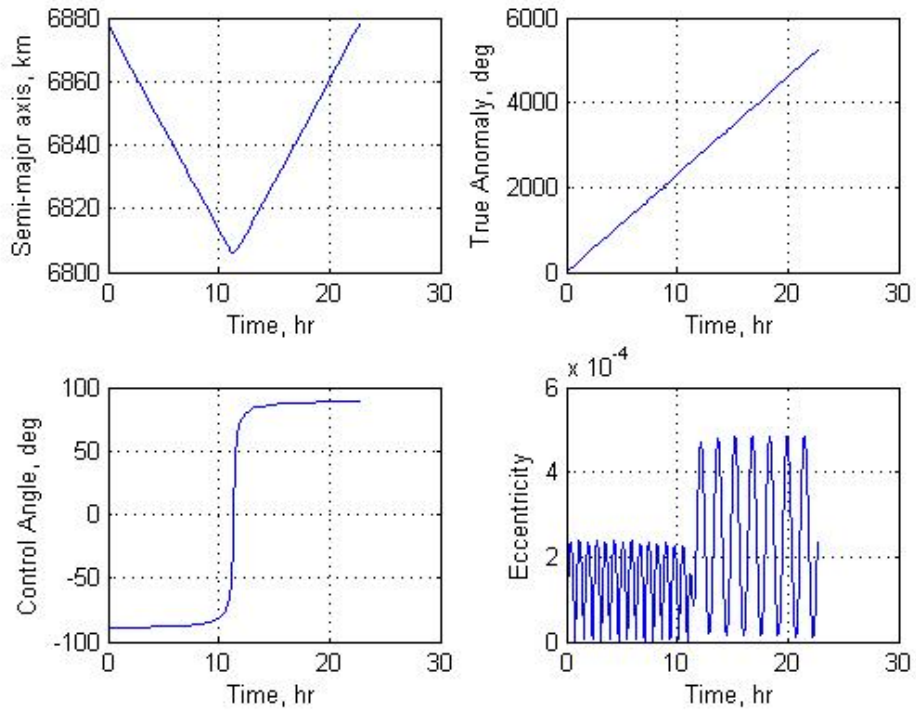

Figure 4.6: Optimal Solution for Test Case 3

starts at -90^o and ends at 90^o as shown in the bottom left plot. The assumption that the eccentricity remains close to zero is once again validated in the bottom right plot. The top left plot shows that, with the longer maneuver duration, the spacecraft was able to lower its altitude by about $73km$. The duration of the maneuver for this case is $81,655sec$, or $22.7hr$, and the corresponding ΔV is $0.082km/s$. The additional Lagrange cost is 18.1630, which is larger than the Lagrange cost from the previous two cases but still orders of magnitude smaller than the Mayer cost. Figure 4.7

shows the final orbit ground track of the reference spacecraft and depicts the position of the maneuvering spacecraft at the end of the maneuver, and the corresponding position of the spacecraft on the reference orbit. The position markers in Figure

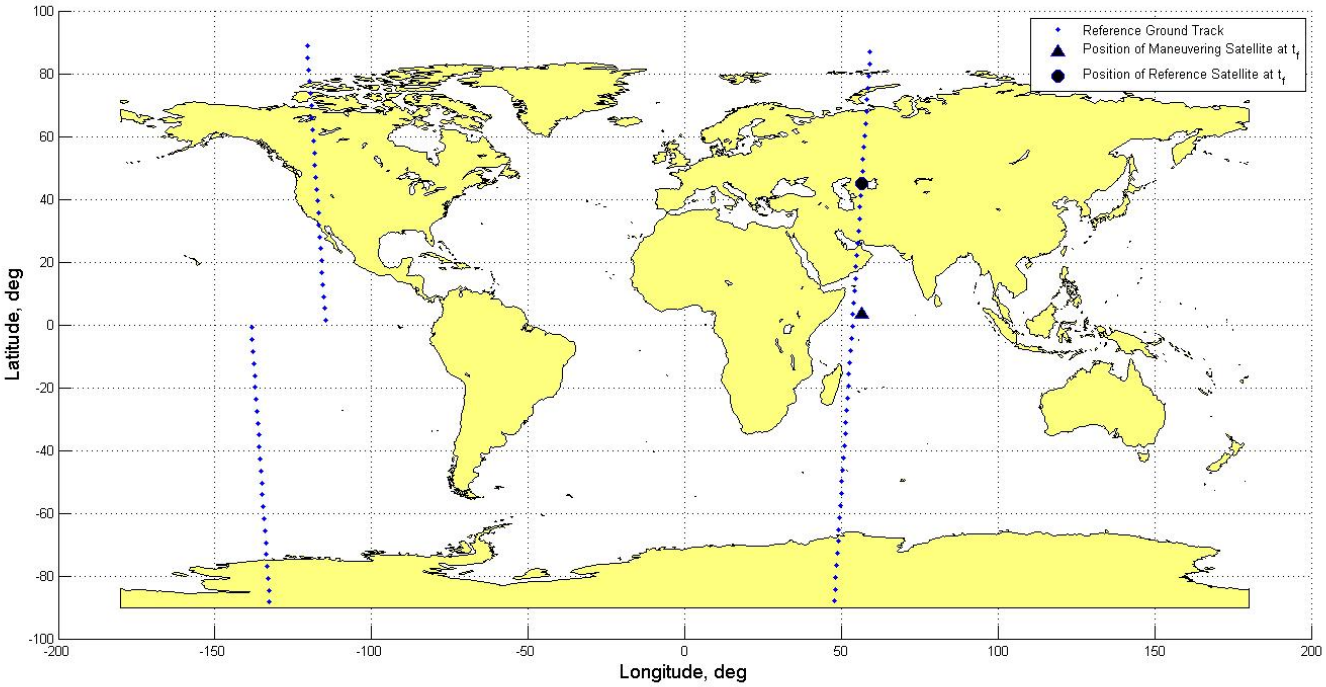

Figure 4.7: Ground Track for Test Case 3

4.7 show a significant separation between the maneuvering and reference spacecraft after maneuvering for almost 23 hours. The Δt for this case is $11min$, which is a substantial change for the relatively low ΔV expended. Figure 4.7 also shows that at the end of the maneuver the spacecraft is slightly offset from the reference ground track. This eastward shift is due to the increase in velocity while the spacecraft's altitude is lowered. This effect would need to be accounted for when solving the full target overflight problem, but does not affect the results of this research.

4.1.4 Test Case 4. The fourth case uses the same initial conditions and target position as Case 3 (Section 4.1.3), but removes the final altitude constraint on the spacecraft. Without the altitude constraint, the spacecraft will be able to lower

its altitude and reach the target position as fast as possible, therefore maximizing the value of Δt. Since the control angle is expected to be constant, α can be set to zero for this case. Figure 4.8 shows the optimal solution for this scenario. The

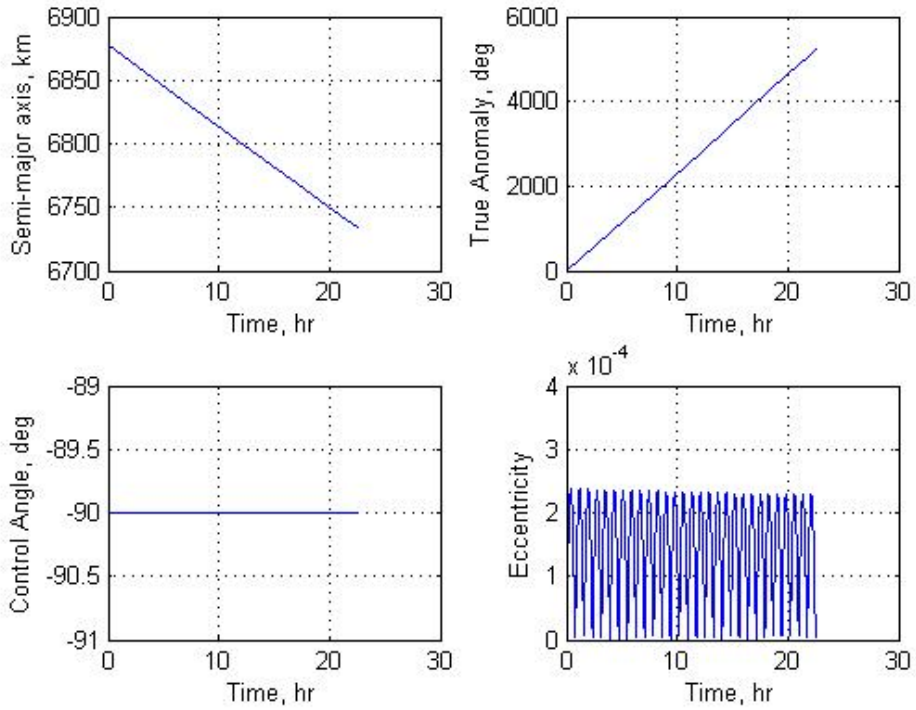

Figure 4.8: Optimal Solution for Test Case 4

bottom left plot of Figure 4.8 shows that the control angle for this case is constant at -90^o, meaning that the spacecraft thrusts continuously in the anti-velocity direction to lower its altitude. The top left plot shows that the altitude loss for this case is about $144km$, which is almost double the altitude loss in Case 3. The duration of the maneuver for this case is $81,011sec$, or $22.5hr$, and the corresponding ΔV is $0.081km/s$. Since $\alpha = 0$ for this case, there is no additional Lagrange cost. Figure 4.9 shows the final orbit ground track of the reference spacecraft and depicts the position of the maneuvering spacecraft at the end of the maneuver, and the corresponding position of the spacecraft on the reference orbit. The position markers in Figure 4.9 show an even larger separation than the one seen in Case 3. The Δt for this case is

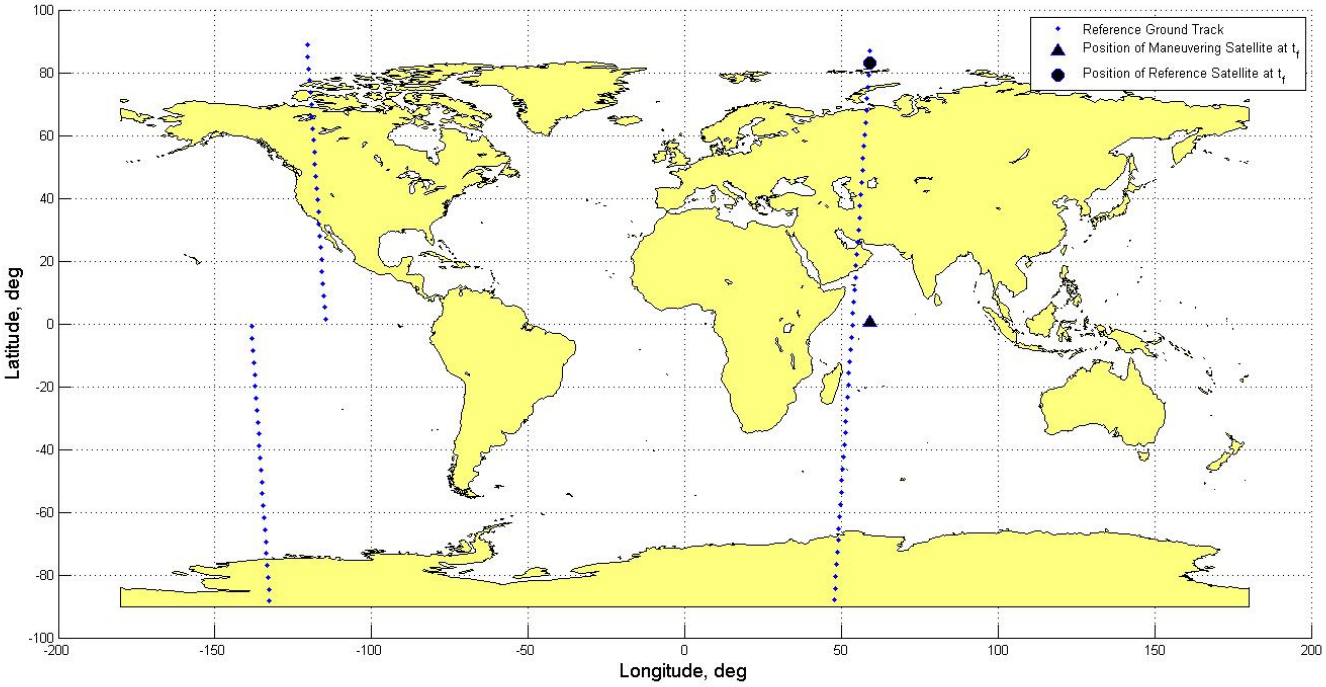

Figure 4.9: Ground Track for Test Case 4

21.7min, which is almost double the Δt seen in Case 3. The ground track shift effect that was discussed in Case 3 is also visible in Figure 4.9.

4.1.5 Summary of Optimal Control Results. The preceding sections presented results for four in-plane maneuvers using optimal control. The results from the four cases are tabulated in Table 4.1 for the reader's convenience.

Table 4.1: Optimal Control Results

Case No.	a_f Constraint	ν_f, deg	t_f, sec	ΔV, km/s	Δt Created
1	Yes	180	3152.5	0.0032	0.97 sec
2	Yes	900	14,172	0.014	20 sec
3	Yes	5220	81,655	0.082	11 min
4	No	5220	81,011	0.081	21.7 min

The results in Table 4.1 show that constraining the final altitude of the spacecraft can decrease the amount of Δt created by the maneuver. Unless specified in the

mission requirements, the final altitude of the spacecraft should be left unconstrained for the types of maneuvers discussed. As a result, in order to maximize Δt, spacecraft should maneuver by thrusting continuously in the anti-velocity direction.

4.2 Feedback Control Results

The feedback control algorithm is used to demonstrate both in-plane and out-of-plane maneuverability. The in-plane results are compared to the optimal control result for Case 4 presented in Section 4.1.4. The out-of-plane maneuverability is demonstrated by performing an inclination change maneuver. The results of the maneuver are compared to the optimal results presented in Alfano's thesis [24]. A combined semi-major axis and inclination maneuver is also analyzed to examine the effect of out-of-plane maneuvering on Δt. Finally, the implementation of the thrust-coast duty cycle is demonstrated by repeating the inclination change maneuver.

4.2.1 In-Plane Maneuver. The spacecraft starts at the same conditions as in Case 4 (Section 4.1.4) - circular polar orbit at an altitude of $500km$. The position of the target is chosen at $\nu_f = 5220^o$, as in Section 4.1.4. The gains in the Lyapunov function for this case are $K_1 = K_\nu = 1$ and $K_2 = K_3 = K_4 = K_5 = K_6 = 0$. Figure 4.10 shows a plot of the in-plane and out-of-plane control angles computed by the feedback control algorithm. Figure 4.10 shows that the in-plane angle is constant at -90^o, and the out-of-plane, which is not clearly visible in the plot, is constant at 0^o. In this case, the feedback controller produced a result that is exactly the same as the optimal result in Section 4.1.4. Figure 4.11 shows plots of all six COEs during the maneuver. The top two plots in Figure 4.11 show the change in semi-major axis and eccentricity, and their behavior is identical to the optimal results seen in Section 4.1.4. The middle two plots show that the inclination and RAAN are constant, since the maneuver is coplanar. The plot of the true anomaly, or argument of latitude for circular orbits, shows that the spacecraft ends at the desired final position. Figure 4.12 shows the final orbit ground track of the reference spacecraft and depicts the position

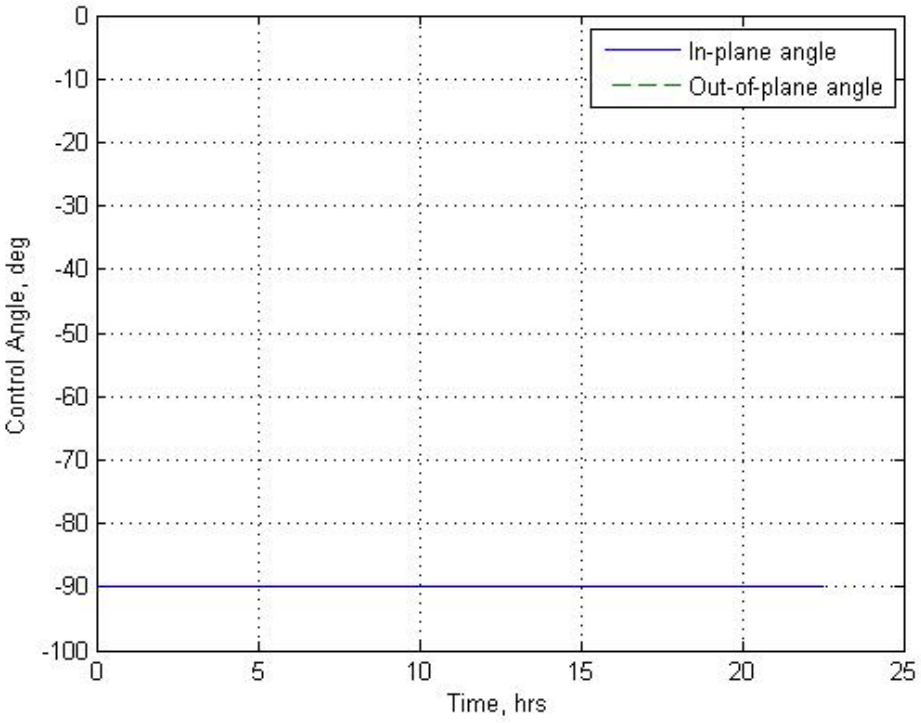

Figure 4.10: Control Angles for In-plane Maneuver Using Feedback Control

of the maneuvering spacecraft at the end of the maneuver, and the corresponding position of the spacecraft on the reference orbit. Figure 4.12 is exactly the same as the ground track for the optimal maneuver depicted in Figure 4.9. Therefore the Δt created in this case is $21.7min$, which is equivalent to the optimal maneuver result.

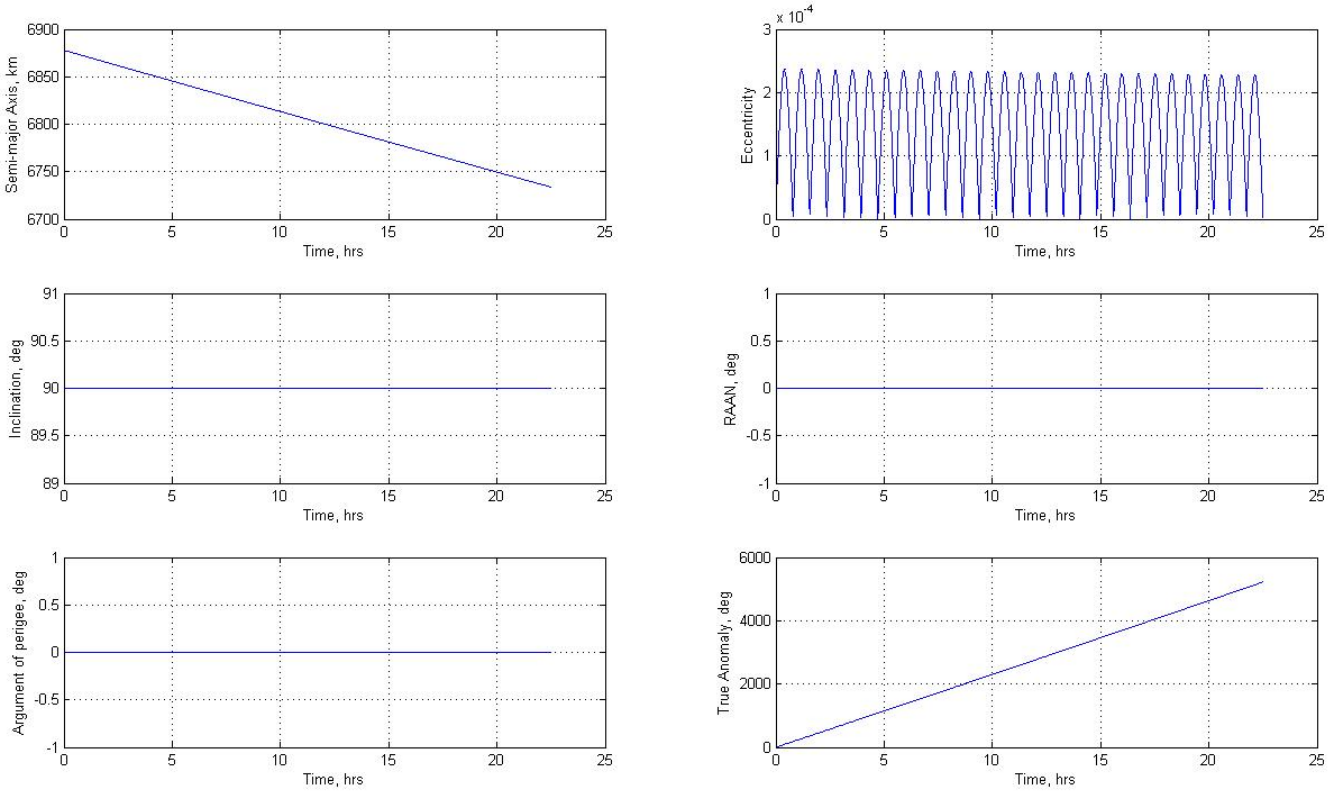

Figure 4.11: COEs for In-plane Maneuver Using Feedback Control

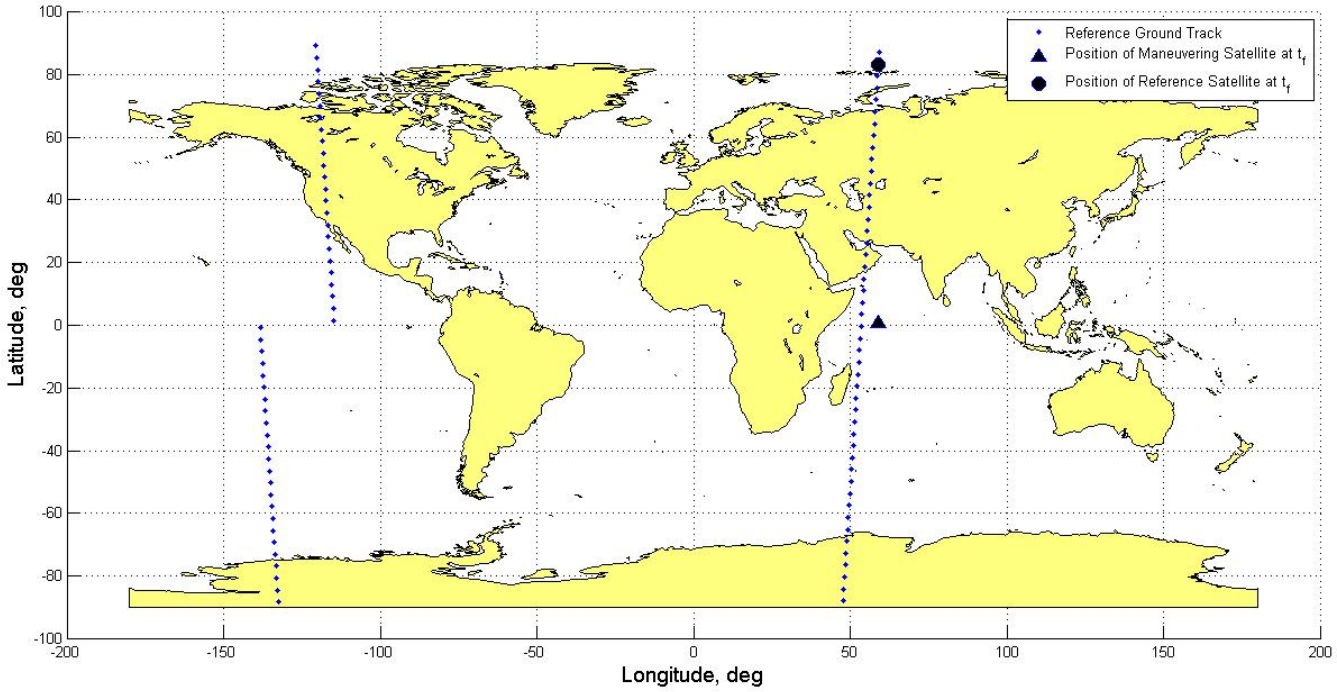

Figure 4.12: Ground Track for In-plane Maneuver Using Feedback Control

4.2.2 Out-of-Plane Maneuver. A spacecraft in circular orbit at an altitude of $500km$, inclined at 45^o is considered. With $K_3 = 1$ and all other gains set to zero the feedback controller is used to achieve an inclination of 40^o, and the solution for one orbital period is output. Figure 4.13 shows the control angles computed by the feedback control algorithm. The plot in Figure 4.13 shows that the in-plane control

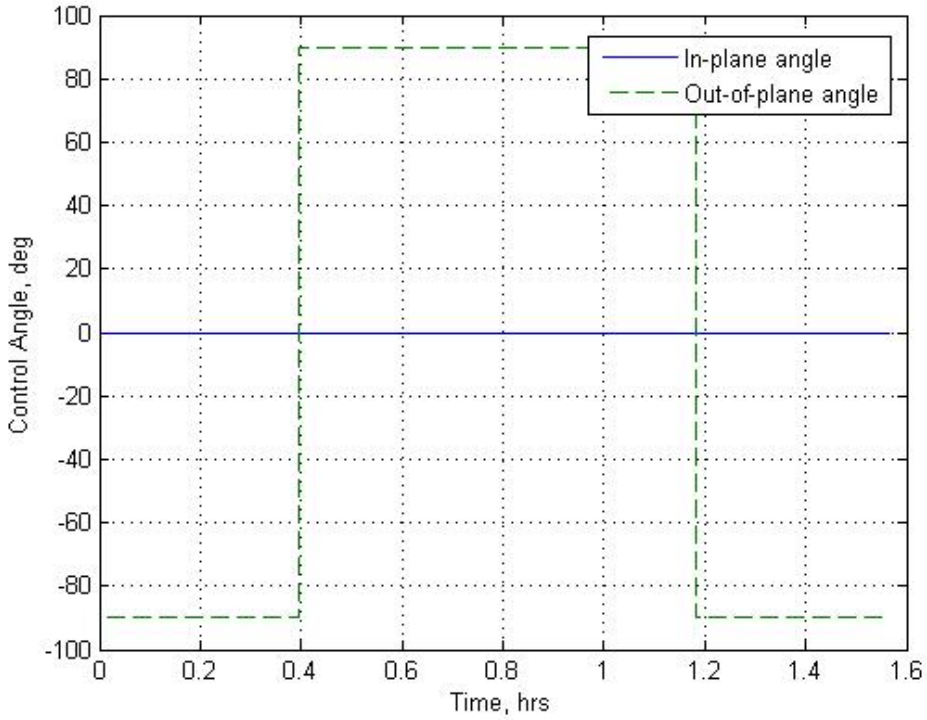

Figure 4.13: Control Angles for Inclination Change over One Orbit

angle is constant at 0^o, which makes sense since only the out-of-plane component of acceleration acts to change inclination. The out-of-plane control angle starts at -90^o, switches to 90^o about a quarter way through the orbit, and switches back to -90^o for the last quarter of the orbit. The result in Figure 4.13 agrees with the optimal result presented by Alfano [24]. Closer examination of Equation 2.30, defining the rate of change of the inclination, shows that inclination changes as a function of $\cos\nu$. Therefore, the change in thrust direction is necessary to account for the sign change in the cosine function, and accumulate inclination change. If the thrust direction had not changed throughout the orbit the net inclination change would be zero.

Figure 4.14 shows plots of all six COEs throughout the maneuver. Since only

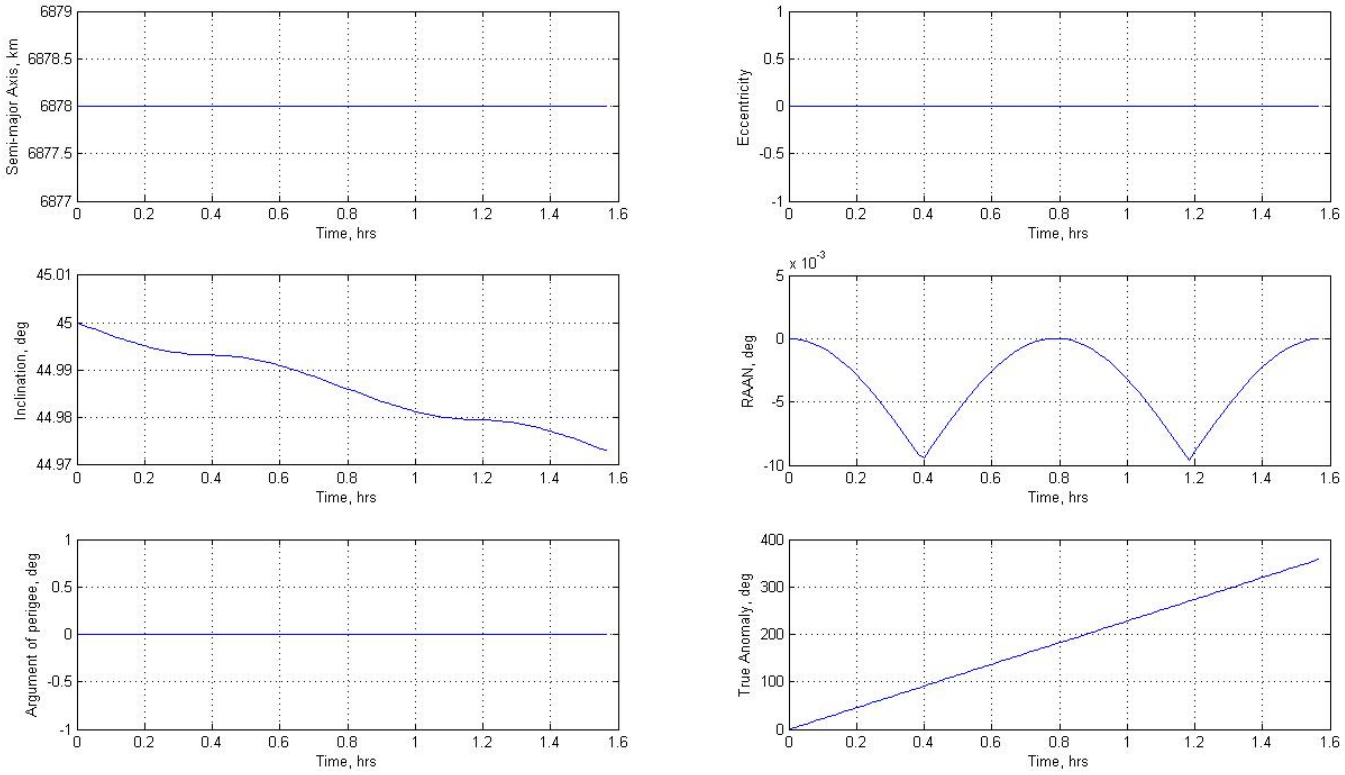

Figure 4.14: COEs for Inclination Change Maneuver over One Orbit

an out-of-plane acceleration component is applied the only COEs that change, other than change in ν due to the mean motion, are the inclination and RAAN, shown in the middle two plots of Figure 4.14. The left plot shows that the inclination at the end of one orbit is about 44.97^o, which also agrees with Alfano's analytical result [24]. The plot on the right shows that RAAN changes, but does not accumulate change, like the inclination, because RAAN actually changes as a function of $\sin i$, as defined in Equation 2.31.

In this case, since the semi-major axis of the osculating orbit does not change, and as a result the mean motion remains constant, the spacecraft does not gain any Δt. In order to explore the possibility of affecting Δt with out-of-plane maneuvers, a combined semi-major axis and inclination change maneuver is considered. As ex-

plained in Section 3.2.2 the mean motion control method presented by Naasz should not be used until the desired inclination is reached. Therefore, mean motion control is not effective for this case. In order to perform the combined semi-major axis and inclination change maneuver the controller is asked to decrease the semi-major axis and inclination by an arbitrary amount and is allowed to run for $22.5hr$ - the maneuver duration for the in-plane maneuver case in section 4.2.1, and therefore expends the same amount of ΔV. The result of this maneuver is then compared to the in-plane maneuver result.

The in-plane maneuver presented in Section 4.2.1 is reconsidered with the additional requirement of accumulating inclination change. The gain values K_1 and K_3 are initially set to unity and all other gains are set to zero. Two more gain combinations are used to examine the difference in the results due to different gain values ($K_1 = 10, K_3 = 100$ and $K_1 = 100, K_3 = 10$). Figure 4.15 shows the results for all three sets of gain values. The in-plane angle for all three cases was constant at -90^o and is not shown in the plots. For the first case, $K_1 = 1, K_3 = 1$, the semi-major axis decreases to $6743km$ and the inclination decreases to 89.86^o. The out-of-plane control angle alternates between -90^o to 90^o, as shown in the bottom right plot of Figure 4.15. Based on the final value of ν from the bottom left plot, the Δt achieved in this case is $20.6min$, which is over one minute less than the in-plane maneuver result. For the second case, $K_1 = 10, K_3 = 100$, the controller focuses on changing the inclination more than the semi-major axis. As a result, the semi-major axis decreases to $6803km$ and the inclination decreases to 89.65^o. The out-of plane control angle alternates between -90^o and 90^o, as in the first case, but a closer look at the plot reveals that the amount of time spent thrusting directly out-of-plane is longer in this case. Based on the final value of ν the Δt achieved in this case is $11min$, which is significantly less than the in-plane maneuver result. Finally, for the last case, $K_1 = 10, K_3 = 100$, the controller focuses on changing the semi-major axis more than the inclination. As a result, the semi-major axis decreases to $6734km$ and the inclination decreases to 88.98^o. The out-of-plane-control angle, in this case, starts oscillating between -20^o

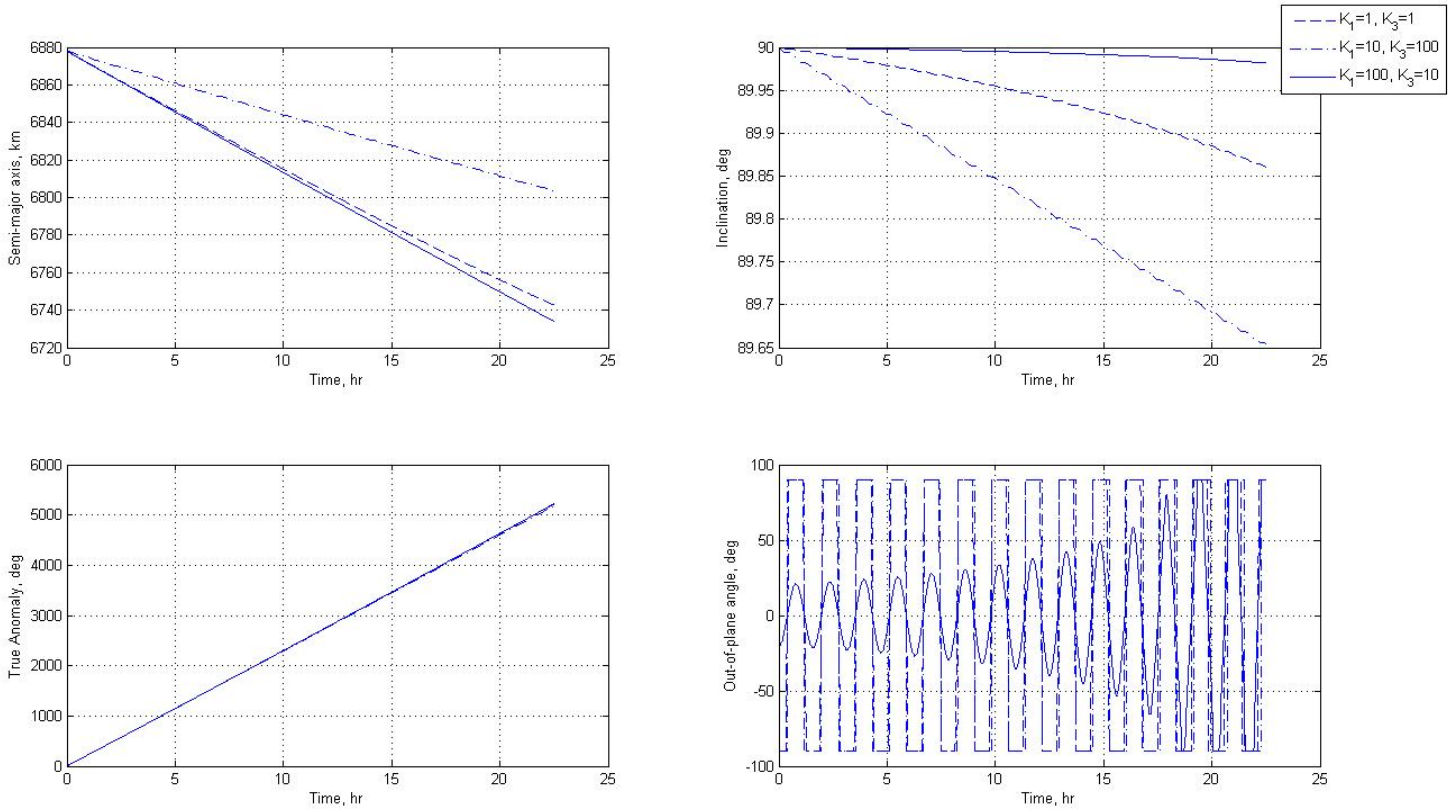

Figure 4.15: Combined Semi-major axis and Inclination Change Maneuver

and 20^o and gradually increases in amplitude to approach $\pm 90^o$. Based on the final value of ν the Δt achieved in this case is $21.5 min$, which is just slightly less than the in-plane result.

The results presented above show that with the additional requirement to change inclination, the spacecraft was not able to achieve the same amount of Δt as with the in-plane maneuvers. Therefore, in-plane maneuvers provide the maximum amount of Δt. Out-of-plane maneuvers should only be considered if mission requirements dictate an orbit plane change.

4.2.3 Thrust-Coast Duty Cycle Implementation. The ability to implement a thrust-coast duty cycle is demonstrated by repeating the inclination change maneuver over one orbital period, presented in Section 4.2.2. In this case, however, the

spacecraft is only allowed to thrust continuously for 10 seconds, and must coast for 2 minutes between pulses. The 10 seconds on-2 minutes off cycle is selected arbitrarily to demonstrate the algorithm's capability, and the two numbers defining the duty cycle can be easily changed to meet any user requirement. Figure 4.16 shows the control angles computed by the feedback algorithm. Figure 4.16 clearly depicts the

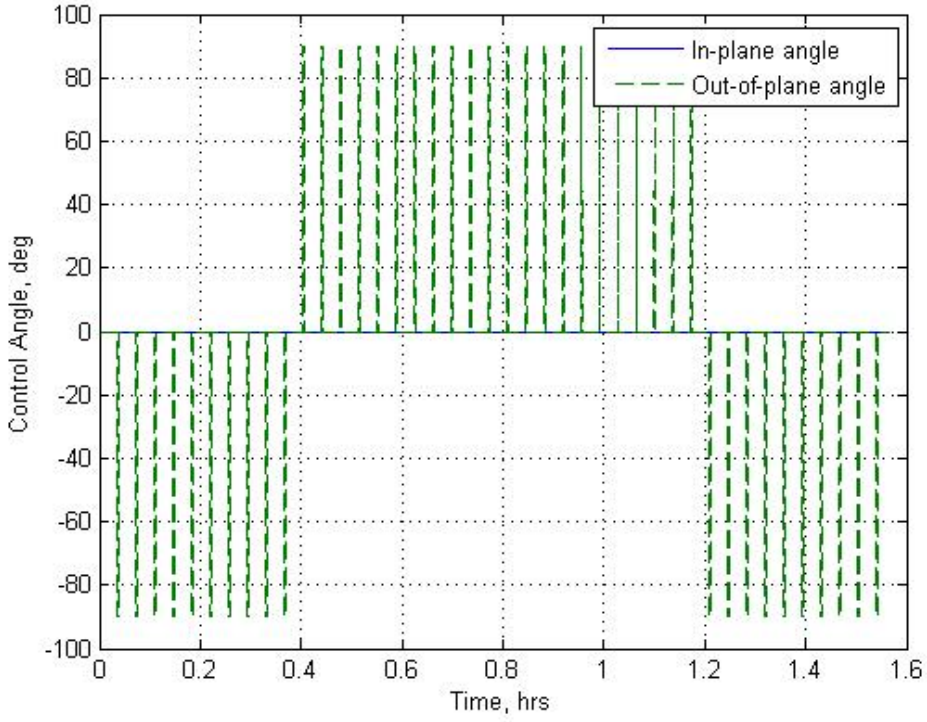

Figure 4.16: Control Angles for Inclination Change with Thrust-Coast Duty Cycle

duty cycle, by showing the out-of-plane control angle at $\pm 90^o$ in 10 second increments when the spacecraft is thrusting, and at 0^o for 2 minute increments when the spacecraft is coasting. It is importnat to note that the spikes in Figure 4.16 represent the time intervals that the control vector magnitude is non-zero, and the spaces between spikes represent time intervals when the control vectror magnitude is zero (coasting). The spikes do not nessarily mean that the actual thruster on a spacecraft must return to 0^o during the coasting periods.

Figure 4.17 shows plots of all six COEs throughout the maneuver. The middle

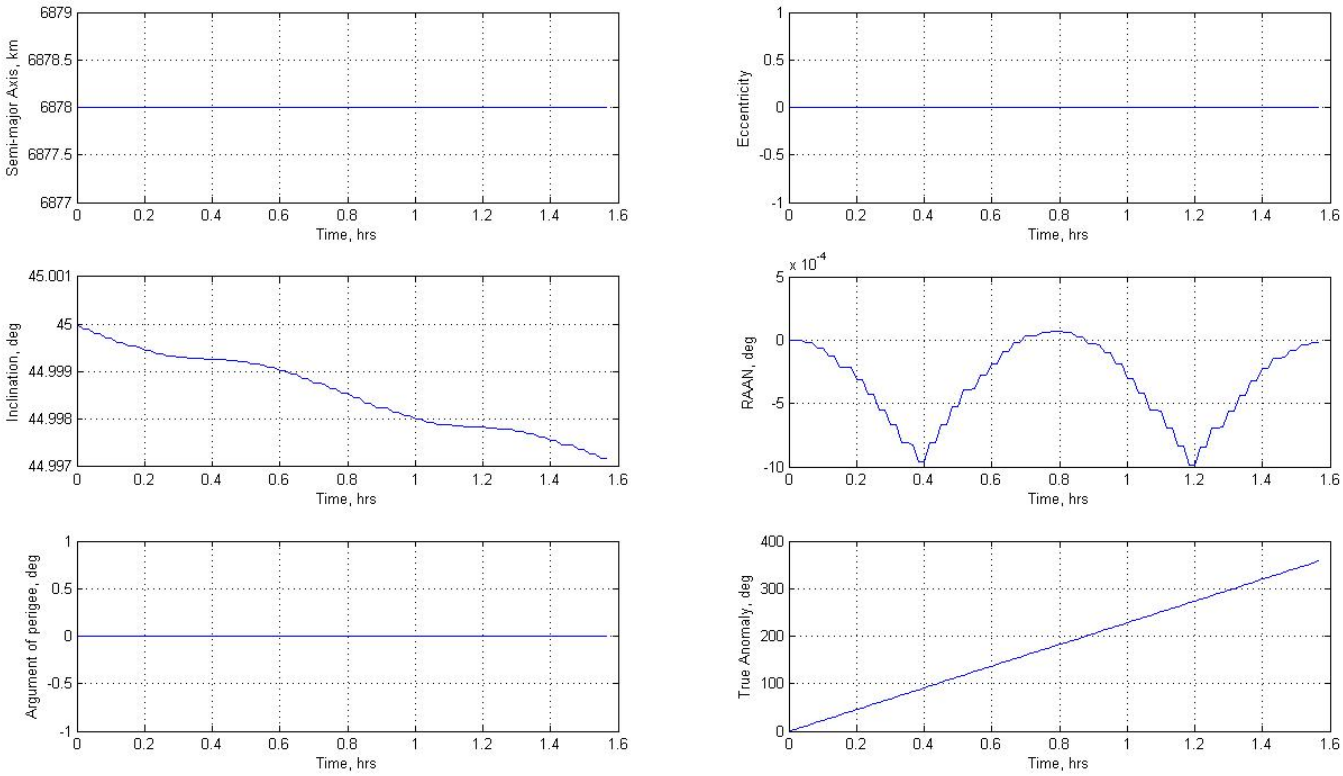

Figure 4.17: COEs for Inclination Change with Thrust-Coast Duty Cycle

left plot in Figure 4.17 shows that the inclination change over one orbit in this case is much less than the inclination change for the continuous thrusting case. This result is intuitive since the spacecraft thrusts for a significantly lower amount of time throughout the maneuver.

The thrust-coast duty cycle can also be implemented by calculating an acceleration magnitude, lower than the maximum acceleration magnitude available from the propulsion system, that can be used in the continuous thrusting algorithm to produce the same result. The amount of ΔV added per pulse can be calculated by:

$$\Delta V_{pulse} = T_{on} A \tag{4.1}$$

where T_{on} is the duration of the pulses, and A is the maximum acceleration magnitude available from the propulsion system. The total number of pulses throughout the maneuver is given by:

$$n = \frac{t_f}{T_{on} + T_{off}} \tag{4.2}$$

where t_f is the duration of the maneuver, T_{on} is the duration of the pulses, and T_{off} is the duration of the coasting period between pulses. The total ΔV added during the maneuver can then be written as:

$$\Delta V_{tot} = n\Delta V_{pulse} = A_{approx} t_f \tag{4.3}$$

where A_{approx} is the acceleration magnitude that can be used in the continuous thrusting algorithm to produce the results of the thrust-coast duty cycle. Solving equation 4.3 for A_{approx} provides the result:

$$A_{approx} = \frac{T_{on}}{T_{on} + T_{off}} A \tag{4.4}$$

Equation 4.4 provides a significant result by showing that the implementation of the thrust-coast duty cycle does not necessitate the development of a new algorithm. It can be simulated in the continuous thrusting algorithm by modifying the acceleration magnitude appropriately.

4.2.4 Summary of Feedback Control Results. The results presented in the preceding sections demonstrated in-plane and out-of-plane maneuvers using a feedback control algorithm. The result of the in-plane maneuver was exactly equivalent to the optimal result presented in Section 4.1.4. An inclination change maneuver was presented to examine the inclination change over one orbital period and the result matched the optimal control result presented by Alfano [24]. A combined semi-major axis and inclination change maneuver was then presented and showed that maneuvering out-of-plane only decreases Δt, and, unless absolutely necessary, should be avoided. Finally, the thrust-coast duty cycle implementation was demonstrated by

repeating the previous inclination change maneuver, and it was shown that the duty cycle can be simulated in the continuous thrusting algorithm by adjusting the value of the constant acceleration magnitude.

4.3 Analytical Approach

Sections 4.1 and 4.2 presented intuitive results for in-plane maneuvers. It was shown that the maximum amount of Δt is achieved by thrusting continuously against the velocity direction. This result can be used to derive an analytical equation for the amount of Δt created during any in-plane maneuver. Recall the two first-order differential equations used in setting up the optimal control problem for in-plane maneuvers, equation 3.8 and 3.9, with $\theta = -90^o$:

$$\frac{da}{dt} = -\frac{2}{\sqrt{\mu}} a^{3/2} A \tag{4.5}$$

$$\frac{d\nu}{dt} = \sqrt{\frac{\mu}{a^3}} \tag{4.6}$$

Equation 4.5 can then be integrated analytically after separating variables, as also shown by Wiesel [14]:

$$\int_{a_0}^{a_f} a^{-3/2} da = \int_{t_0}^{t_f} -\frac{2}{\sqrt{\mu}} A dt \tag{4.7}$$

Integrating both sides of the equation above provides a result for the duration of the maneuver:

$$t_{man} = t_f - t_0 = \frac{\sqrt{\mu}}{A} \left(a_f^{-1/2} - a_0^{-1/2} \right) \tag{4.8}$$

The only unknown value in equation 4.8 is the final value of the semi-major axis, a_f. Consider a change of independent variables from t to ν, yielding a new differential equation:

$$\frac{da}{d\nu} = \frac{da}{dt}\frac{dt}{d\nu} = -\frac{2}{\mu} a^3 A \tag{4.9}$$

By separating variables the equation can be integrated analytically:

$$\int_{a_0}^{a_f} a^{-3} da = \int_{\nu_0}^{\nu_f} -\frac{2}{\mu} A d\nu \tag{4.10}$$

which provides an expression for a_f in terms of ν_f, which is defined:

$$a_f = \left(\frac{1}{a_0^2} + \frac{4}{\mu} A(\nu_f - \nu_0) \right)^{-1/2} \tag{4.11}$$

Equation 4.11 can then be substituted into equation 4.8 and the time of arrival difference, Δt, can be calculated by:

$$\begin{aligned} \Delta t =& (\nu_f - \nu_0) \sqrt{\frac{a_0^3}{\mu}} - t_{man} = \\ =& (\nu_f - \nu_0) \sqrt{\frac{a_0^3}{\mu}} - \frac{\sqrt{\mu}}{A} \left[\left(\frac{1}{a_0^2} + \frac{4}{\mu} A(\nu_f - \nu_0) \right)^{1/4} - a_0^{-1/2} \right] \end{aligned} \tag{4.12}$$

The first term in equation 4.12 represents the amount of time it would take the spacecraft to reach ν_f without maneuvering (the reference spacecraft), and the second term is the duration of the maneuver. The analytic equation for Δt can be a powerful tool in analysis and mission planning.

The Δt expression is used to examine how different values of the initial semi-major axis affect the maneuvers. Given a range of values for ν_f, the expressions presented above are used to calculate the maneuver duration and Δt for two different values of a_0. Figure 4.18 shows the resulting plot. Although the two lines in Figure 4.18 are very close to each other, it can be easily seen that a spacecraft in a higher orbit that maneuvers for the same amount of time as a spacecraft in a lower orbit, thus burning the same amount of fuel, will achieve slightly higher values of Δt. This result is directly related to the fact that a spacecraft starting in a higher orbit will lose a larger percentage of its altitude than a spaceraft starting in a lower orbit after maneuvering for the same amount of time. The plot shown in Figure 4.18 can be

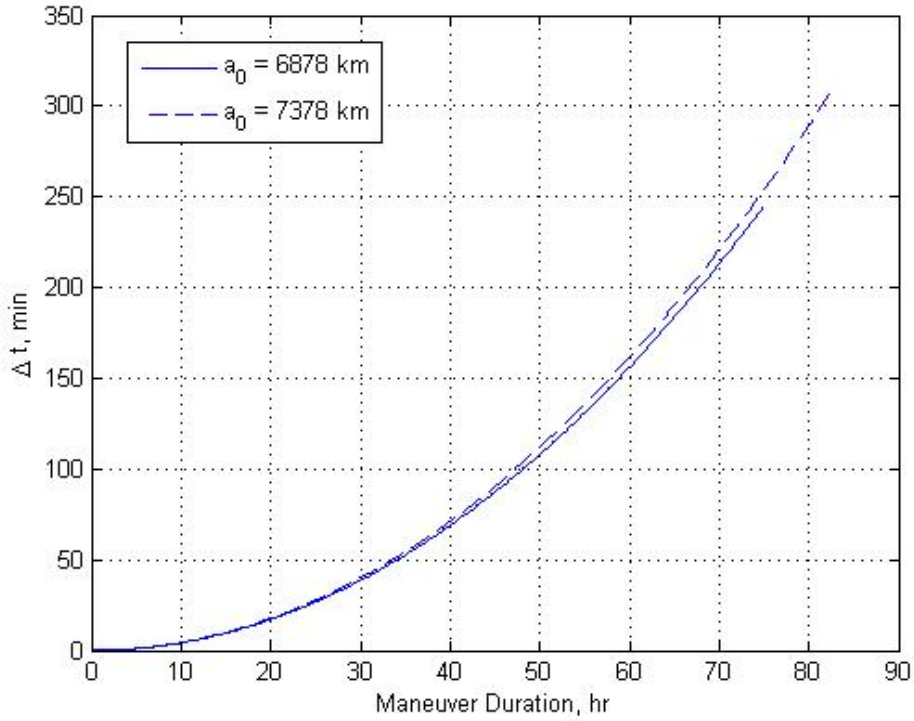

Figure 4.18: Δt as a function of Maneuver Duration

useful when planning missions, and is a basic example of how the Δt expression can be used to provide valuable information.

4.4 Spacecraft Maneuver Simulation

In Section 4.3 an expression for Δt was derived. This section uses the analytical expression, combined with the feedback control algorithm that was developed in Section 3.2, to analyze a spacecraft maneuver that resembles a real world scenario more closely. The subject spacecraft is in circular orbit at a given altitude, $a_0 = 7378km$, inclined at $i = 45^o$, and is tasked to maneuver, expending the minimum amount of fuel, within a given amount of lead time, t_{lead}, so that the final position, ν_f, is reached 30 seconds earlier than expected, or $\Delta t = 30sec$ at the end of the maneuver. The initial position of the spacecraft is set to $\nu_0 = 0^o$. Depending on the amount of lead time available, the spacecraft may need to thrust for the entire time, thrust for a cer-

tain amount of time and drift for the remainder, or provide feedback that the desired Δt is not achievable in the amount of time given. A term can be added to the Δt expression, equation 4.12, that accounts for the drifting time as such:

$$\Delta t = (\nu_f - \nu_0)\sqrt{\frac{a_0^3}{\mu}} - t_{man} - t_{drift} \tag{4.13}$$

where $t_{man} + t_{drift} = t_{lead}$. Equation 4.13 can then be solved for ν_f, which defines the final position within the orbit:

$$\nu_f = \nu_0 + (\Delta t + t_{lead})\sqrt{\frac{\mu}{a_0^3}} \tag{4.14}$$

The spacecraft then starts at ν_0, maneuvers to an intermediate position, ν_1, and drifts to the final position ν_f. Based on equations 4.8 and 4.11 the duration of the maneuver and the final value of the semi-major axis can be written as:

$$t_{man} = \frac{\sqrt{\mu}}{A}\left[\left(\frac{1}{a_0^2} + \frac{4}{\mu}A(\nu_1 - \nu_0)\right)^{1/4} - a_0^{-1/2}\right] \tag{4.15}$$

$$a_f = \left(\frac{1}{a_0^2} + \frac{4}{\mu}A(\nu_1 - \nu_0)\right)^{-1/2} \tag{4.16}$$

t_{drift} is then defined by the mean motion of the orbit reached at the end of the maneuvering period:

$$t_{drift} = (\nu_f - \nu_1)\sqrt{\frac{a_f^3}{\mu}} \tag{4.17}$$

Combining the results in equations 4.15-4.17 and substituting into equation 4.13 yields the following algebraic expression:

$$\begin{aligned}\Delta t =&(\nu_f - \nu_0)\sqrt{\frac{a_0^3}{\mu}} - \frac{\sqrt{\mu}}{A}\left[\left(\frac{1}{a_0^2} + \frac{4}{\mu}A(\nu_1 - \nu_0)\right)^{1/4} - a_0^{-1/2}\right] - \\ &- (\nu_f - \nu_1)\sqrt{\frac{\left(\frac{1}{a_0^2} + \frac{4}{\mu}A(\nu_1 - \nu_0)\right)^{-3/2}}{\mu}}\end{aligned} \tag{4.18}$$

The only unknown quantity in equation 4.18 is the intermediate position, ν_1, which has a unique solution. An algebraic solver is used to compute the value of ν_1, which provides a solution for the duration of the maneuver and the drift period. If the spacecraft is not given enough lead time, however, equation 4.18 cannot be solved, meaning that the desired Δt cannot be achieved in the allotted lead time.

Once t_{man} and t_{drift} are known, the amount of ΔV required to achieve $\Delta t = 30sec$ is given by:

$$\Delta V_{req} = A t_{man} \tag{4.19}$$

where A is the continuous acceleration magnitude. In the feedback control algorithm, another first-order differential equation is added defining the rate of change of the spacecraft's mass [36]:

$$\dot{m} = -\frac{T}{g_0 I_{sp}} \tag{4.20}$$

where T is the constant thrust magnitude, I_{sp} is the engine's specific impulse, and $g_0 = 9.80655 m/s^2$ is the gravitational acceleration at sea level. Equation 4.20 is numerically integrated along with the EOM to track the spacecraft's mass throughout the maneuver. The amount of ΔV can then be calculated at each iteration through the rocket equation [36]:

$$\Delta V = g_0 I_{sp} \ln \frac{m_0}{m} \tag{4.21}$$

where m_0 is the initial spacecraft mass, and m is the mass at the respective integration step. The I_{sp} value chosen for this simulation is $1500sec$, which is in the middle of the range of I_{sp} values for the type of propulsion system discussed in this research [36]. Once the ΔV reaches the amount of ΔV required, calculated from equation 4.19, the algorithm commands the spacecraft to drift.

The first set of simulations analyze maneuvers to achieve $\Delta t = 30sec$ with $t_{lead} = 3hr, 10hr, 24hr$, and $48hr$ respectively. With $t_{lead} = 3hr$, the simulation stops when the equation solver cannot find a solution for ν_1. The other three simulations provide trajectories for the spacecraft thrusting against the velocity direction for

a certain amount of time and drifting for the remainder. Figure 4.19 shows the output plots for the $t_{lead} = 10hr$ case. The four plots in Figure 4.19 show that the

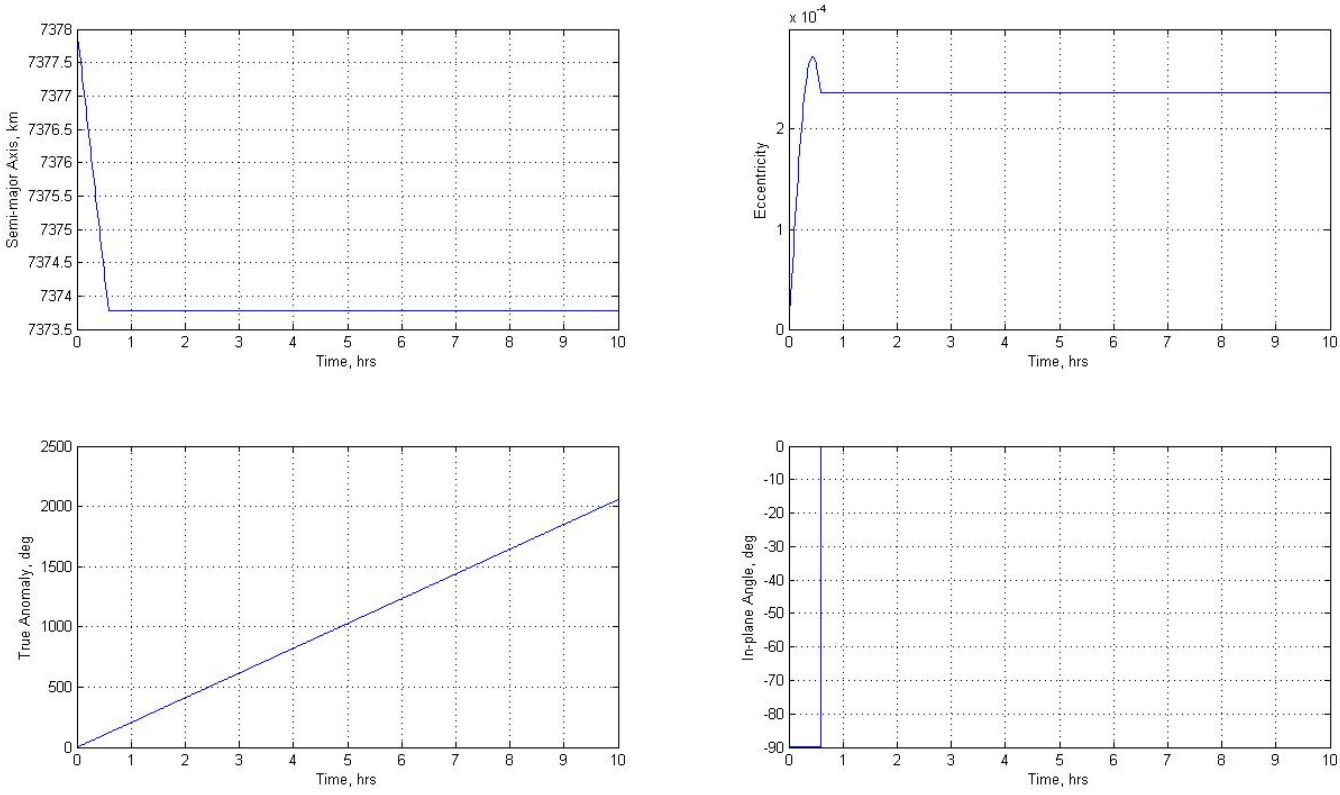

Figure 4.19: Simulation Solution for 10-hour Lead Time

spacecraft thrusts for about $35min$ and drifts for the remainder of the time. While the spacecraft is thrusting in the anti-velocity direction, its semi-major axis decreases and its eccentricity change is negligible. Figure 4.20 shows the final position of the maneuvering spacecraft compared to the ground track of a reference spacecraft. The position markers on Figure 4.20 depict the 30 second separation that is achieved at the end of the maneuver. The plots for the cases where $t_{lead} = 24hr, 48hr$ show the exact same trends, with shorter maneuvering periods and longer drift periods, and are not included.

The results from all four simulations are consolidated and presented in Table 4.2. The results show that the longer the lead time allowed for the maneuver, the

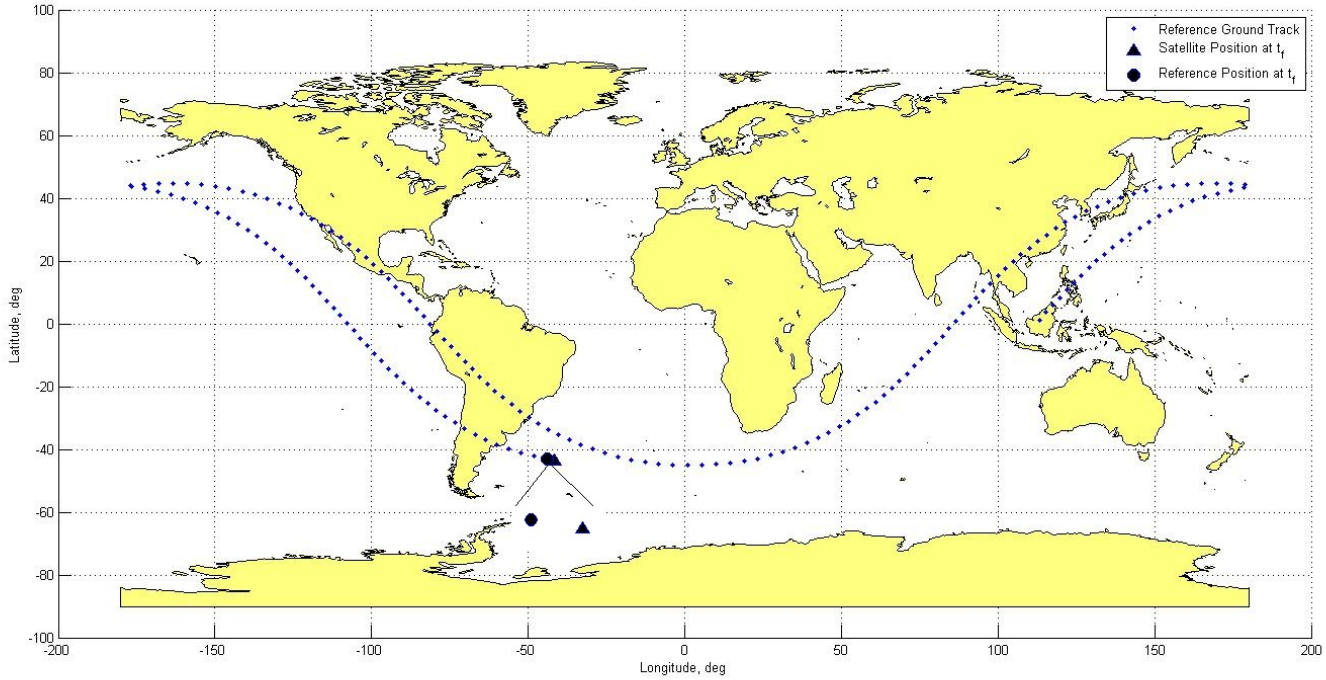

Figure 4.20: Simulation Ground Track for 10-hour Lead Time

Table 4.2: Simulation Results

Sim No.	t_{lead}, hr	t_{man}, min	t_{drift}, hr	ΔV, km/s	a_f, km	Δt, sec
1	3	N/A	N/A	N/A	N/A	N/A
2	10	35.04	9.42	0.0021	7373.8	30
3	24	14.25	23.76	0.00086	7376.3	30
4	48	7.1	47.88	0.00043	7377.1	30

smaller the duration of the thrusting period. Therefore, fuel can be saved by allowing the spacecraft more time to complete the maneuver.

4.4.1 Simulation With Initial Coast Period. Another scenario is considered where the spacecraft has the same objective of $\Delta t = 30sec$, but has an enforced hold time prior to being allowed to maneuver. For example, if the spacecraft generates a trajectory for the maneuver on board, the ground station may need time to analyze the trajectory before allowing the spacecraft to continue. The algorithm then needs to

implement a delay time, t_{delay}, and compute the new initial position of the spacecraft, ν_0^*, after the initial coast period by:

$$\nu_0^* = \nu_0 + t_{delay}\sqrt{\frac{\mu}{a_0^3}} \tag{4.22}$$

Equation 4.18 can be solved, as previously, after replacing ν_0 with ν_0^*. The previous simulation, with $t_{lead} = 10hr$, is rerun with $t_{delay} = 1hr$. Figure 4.21 shows the resulting plots, and clearly depicts the $1hr$ initial delay. The resulting ground track

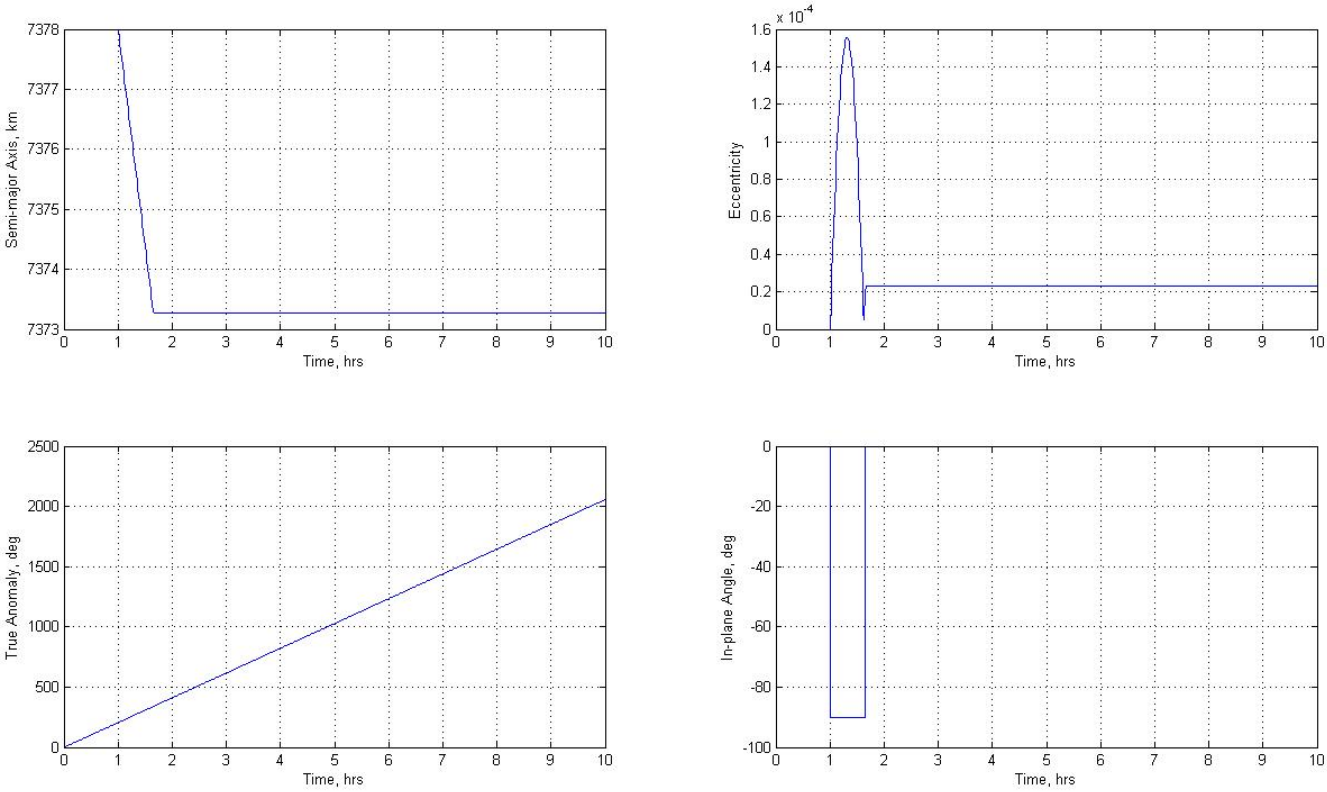

Figure 4.21: Simulation Solution for 10-hour Lead Time with 1-hour Delay

is exactly the same as the ground track shown in Figure 4.20.

With an initial coasting period enforced, the spacecraft must thrust for a longer amount of time, and have less time to drift, to achieve the desired amount of Δt. In this case the spacecraft had to maneuver for $t_{man} = 39.22min$, and drifted for

$t_{drift} = 8.35hr$, corresponding to $\Delta V = 0.002354km/s^2$. This result shows that the initial delay causes the spacecraft to burn more fuel in order to create the same effect. The most efficient way to conduct the maneuver would be to start thrusting as early as possible. However, if an initial delay is necessary, this algorithm could be a useful tool in mission planning.

4.4.2 Simulation With Thrust-Coast Duty Cycle. The last scenario considered includes the implementation of the the 10 seconds on-2 minutes off duty cycle presented in Section 4.2.3. In this case, Equation 4.18 can be solved, as previously, after modifying the acceleration magnitude as shown in Equation 4.4. The previous simulation, with $t_{lead} = 24hr$, is rerun with the thrust-coast duty cycle included. Figure 4.22 shows the resulting plots. The bottom right plot, showing the in-plane

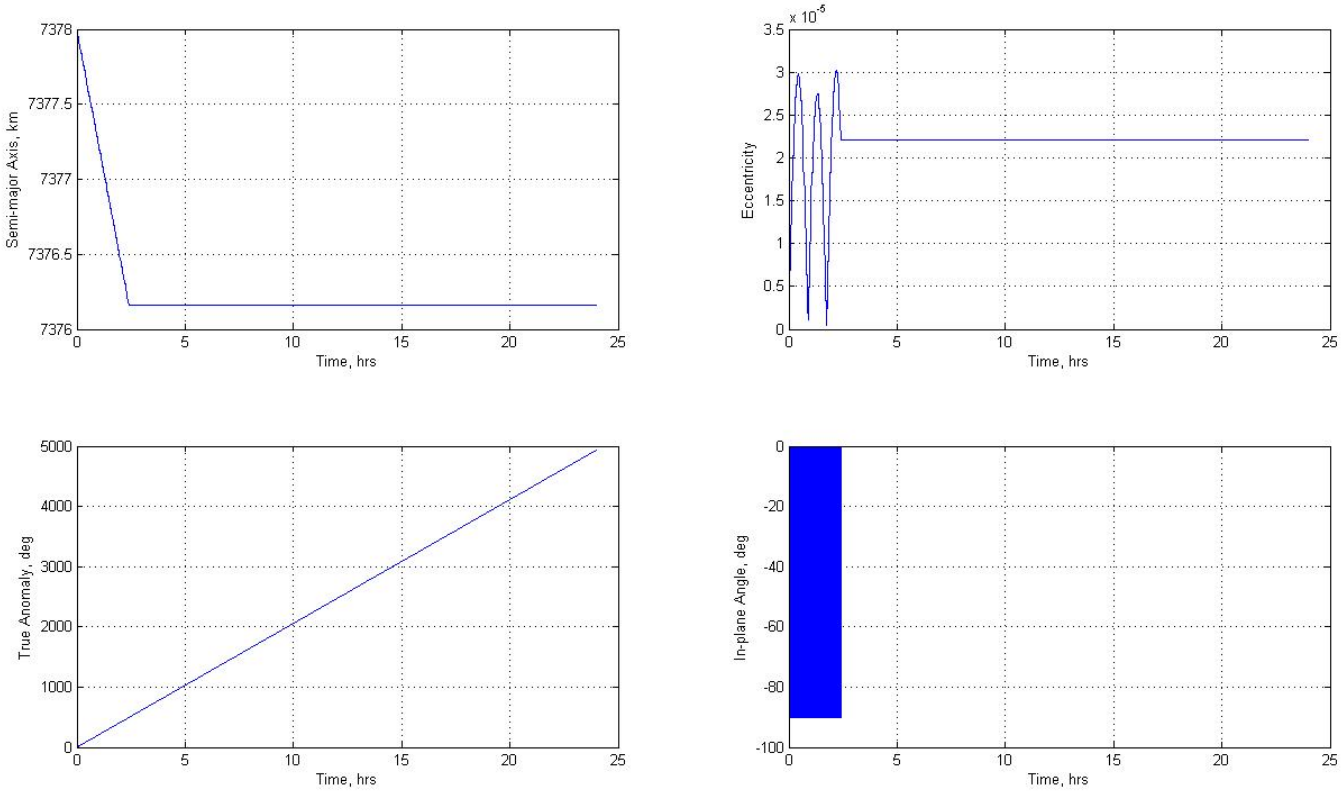

Figure 4.22: Simulation Solution for 24-hour Lead Time with Thrust-Coast Cycle

control angle, in Figure 4.22 depicts the individual 10-second pulses. Due to the large timescale of the plot, however, the individual pulses are not clearly visible. Figure 4.23 shows the resulting ground track, depicting the 30-second separation between the maneuvering and reference spacecraft. As expected, with the thrust-coast duty

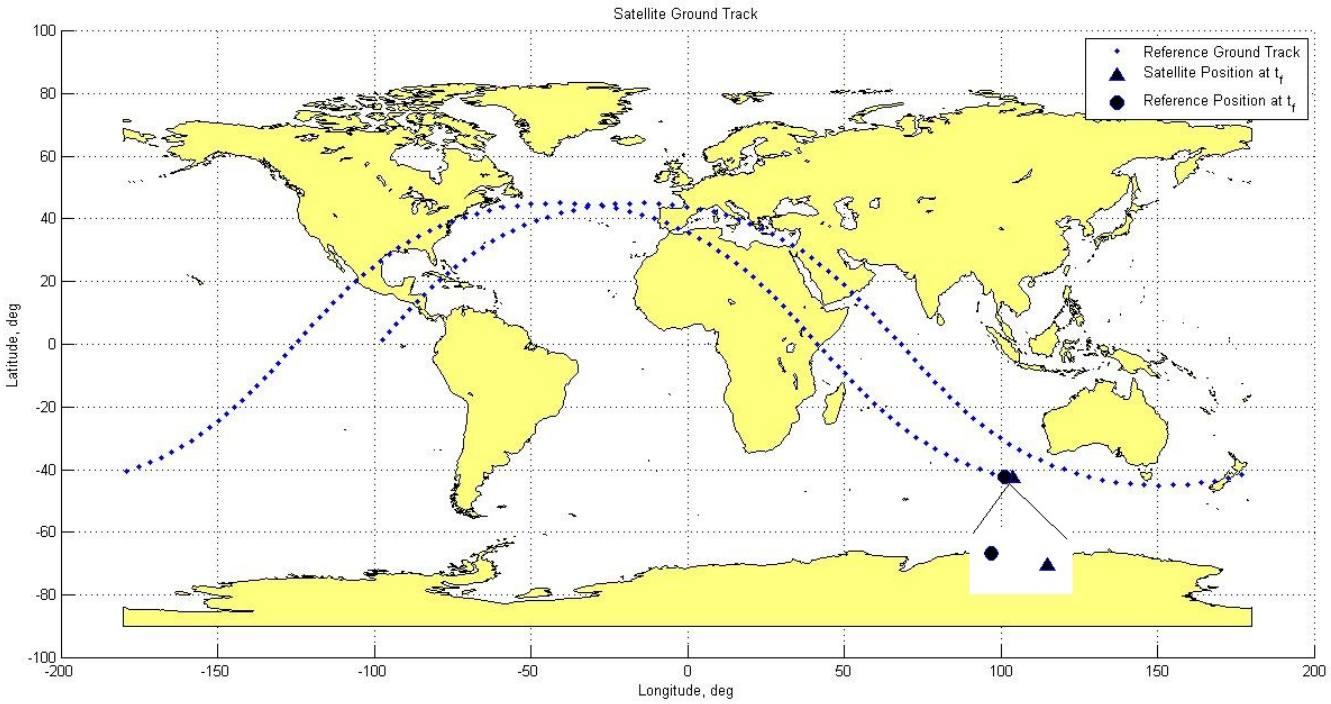

Figure 4.23: Simulation Ground Track for 24-hour Lead Time with Thrust-Coast Cycle

cycle the spacecraft must thrust for a longer amount of time to achieve the desired Δt. In this case the spacecraft had to maneuver for $t_{man} = 3.298hr$, and drifted for $t_{drift} = 20.7hr$. The ΔV expended for this maneuver is $0.000915km/s^2$, which is higher than the result shown on Table 4.2 for the third case because of the longer t_{man} required.

4.5 Summary

This chapter presented the results for several cases analyzed during the course of this research. First, the optimal results were presented for in-plane phasing maneuvers, where the final altitude of the spacecraft is constrained. Then the optimal result for an in-plane maneuver with the final altitude unconstrained was presented, to show that Δt is maximized. Then, an in-plane maneuver using feedback control was presented to show that the result is equivalent to the optimal result. Out-of-plane maneuverability with the feedback control algorithm was then demonstrated, and compared to optimal out-of-plane maneuvers presented by Alfano [24]. These results show that the feedback control algorithm produced optimal results, at least for the cases presented here. Since the optimal control and feedback control algorithms produced the same result, then the feedback control algorithm had a distinct advantage in that it is not computationally intensive and is easy to implement. The feedback control algorithm also allowed for additional capabilities to be added very easily, and the implementation of a thrust-coast duty cycle was demonstrated as an example. Finally, a "realistic" scenario was presented, requiring a spacecraft to maneuver in order to achieve a certain amount of Δt. Results from multiple simulations were presented and compared. It was shown that the most fuel efficient way to conduct the maneuver was to provide as much lead time as possible, and maneuver as early as possible.

V. Conclusions and Future Work

5.1 Overiew

The objective of this research was to develop an algorithm that produces trajectories for responsive maneuvers. The requirement for the maneuver was that the spacecraft had to change its arrival time over a particular location within its orbit, Δt. Two algorithms were developed - an optimal control algorithm using pseudospectral methods, developed through optimal control theory, and a feedback control algorithm, developed through Lyapunov theory. Results using boths algorithms were presented and discussed.

The optimal control algorithm was used to analyze four in-plane maneuvers. The first three maneuvers were basic phasing maneuvers, in which the spacecraft was forced to return to its original altitude. The final maneuver removed the constraint on the final altitude, allowing the spacecraft to remain at a lower altitude at the end of the maneuver. Results showed that the most efficient way to maneuver in-plane was by thrusting in the anti-velocity direction, which was an expected result, and therefore validating optimality. It was also concluded that the amount of Δt is maximized when the final altitude is left unconstrained.

The feedback control algorithm was used to analyze both in-plane and out-of-plane maneuvers. When available, results were compared to optimal solutions to show that feedback control methods can be used to produce optimal results. The in-plane maneuver result was exactly the same as the result shown with the the optimal control algorithm. Out-of-plane maneuverability was demonstrated by an inclination change maneuver and results were compared to optimal results presented by another author. A combined semi-major axis and inclination change maneuver was also presented to show the effects of out-of-plane thrusting on Δt. It was concluded that out-of-plane maneuvering decreases the achievable Δt, and should be avoided unless absolutely necessary.

It was noted that while both algorithms produced the same result for in-plane maneuvers, the feedback control algorithm has a distinct advantage over the optimal

control algorithm. The optimal control algorithm was very sensitive to user-input initial guesses, and failed to converge at times due to numerical difficulties. The feedback control algorithm, on the other hand, was very easy to implement and less computationally intensive. Additionally, it was mentioned that new capabilities and more complexity would be easier to implement in the feedback control algorithm. An example was presented by adding a thrust-coast duty cycle to the algorithm and repeating the inclination change maneuver.

Using the intuitive result that a spacecraft must thrust continuously in the anti-velocity direction to maximize Δt, the simplified equations of motion for the in-plane problem were integrated analytically, and an algebraic expression for Δt was derived. The analytical expression was shown to be a useful result for mission planning purposes. A simple example was presented, where a mission planner may need to decide on the altitude of an orbit. It was shown that a higher orbit would result in higher values of Δt than a lower orbit, after thrusting for the same amount of time.

Finally, using the feedback control algorithm and the derived analytical result, a set of simulations was presented for a "realistic" maneuver. The subject spacecraft was tasked to maneuver, given a certain amount of lead time, to achieve 30 seconds of Δt. Simulation results were presented for different amounts of lead time, for a maneuver start delay, and for a thrust-coast duty cycle system. The results were intuitive, and it was concluded that the most fuel efficient way to conduct this type of maneuver is to allow as much lead time as possible, and begin thrusting as soon as possible.

The development of the two algorithms described throughout this thesis serve as a strong foundation for future algorithms that can be implemented on actual satellites.

5.2 Recommendations for Future Work

Since the focus of this thesis was the development of an algorithm, and not necessarily the analysis of a specific scenario, future work could consider a specific maneuver and provide more insight on the effects of changing different parameters, such as spacecraft mass, thrust magnitude, engine parameters, etc. A few topics were also mentioned throughout this thesis that were not researched in detail. Firstly, the orbit determination and estimation for a spacecraft conducting the maneuvers discussed here would be a necessary research topic in order to fully analyze the effectiveness of responsive flexible collection. Secondly, the full ground target overflight problem needs to be considered next so that the spacecraft actually maneuvers to ovefly a specific point on Earth, instead of an arbitrary position within the orbit. As discussed in Chapter III some work in this area has already been done by other authors, and therefore the focus would be applying those methods to the algorithms developed here. Lastly, the topic of optimal gains for the feedback controller could be researched in order to implement a gain selection process that always produces optimal results. Another important upgrade to the algorithms would be the addition of natural perturbing forces to the equations of motion. For the orbits discussed here, natural effects like air drag and gravitational effects would surely have significant effects, and in some cases could result in fuel savings. The added fidelity to the algorithms would be critical, and analyzing how the addition of perturbing forces changes the maneuver trajectory would be valuable. Chapter II described other sets of variables that can be used to formulate the problem, but only the COEs were considered in this thesis. Reformulating the problem in a different set of variables, namely polar coordinates and equinoctial elements, would provide valuable results. These sets of variables would avoid the singularities encountered in the COE equations and could be used to implement other optimal control or feedback control methods.

Appendix A. The Equations of Motion in Terms of Equinoctial Elements

This appendix is a brief overview of the mathematics behind formulating the equations of motion(EOM) in terms of the equinoctial elements. Although the formulation below has been shown by several authors, the work shown here is taken from [18]. The key terms in formulating the EOM are the partial derivatives of the equinoctial elements with respect to the velocity vector, defined by equation 2.64.

Chobotov first derives the partials of the position vector with respect to the equinoctial elements ($\partial \boldsymbol{r}/\partial a_\beta$) using the definition of the position vector, given in equation 2.58, and the definitions of $\dot{X}_1$ and $\dot{Y}_1$ in equations 2.60 and 2.61.

$$\frac{\partial \boldsymbol{r}}{\partial a} = \frac{\partial X_1}{\partial a}\hat{f} + \frac{\partial Y_1}{\partial a}\hat{g} = \left(\frac{X_1}{a} - \frac{3}{2}\frac{t}{a}\dot{X}_1\right)\hat{f} + \left(\frac{Y_1}{a} - \frac{3}{2}\frac{t}{a}\dot{Y}_1\right)\hat{g} \tag{A.1}$$

$$\frac{\partial \boldsymbol{r}}{\partial h} = \frac{\partial X_1}{\partial h}\hat{f} + \frac{\partial Y_1}{\partial h}\hat{g} \tag{A.2}$$

$$\frac{\partial \boldsymbol{r}}{\partial k} = \frac{\partial X_1}{\partial k}\hat{f} + \frac{\partial Y_1}{\partial h}\hat{g} \tag{A.3}$$

where the partials of the position components with respect to h and k are given by

$$\frac{\partial X_1}{\partial h} = a\left[-(h\cos F - k\sin F)\left(\beta + \frac{h^2\beta^3}{1-\beta}\right) - \frac{a}{r}\cos F(h\beta - \sin F)\right] \tag{A.4}$$

$$\frac{\partial X_1}{\partial k} = -a\left[(h\cos F - k\sin F)\frac{hk\beta^3}{1-\beta} + 1 + \frac{a}{r}\sin F(\sin F - h\beta)\right] \tag{A.5}$$

$$\frac{\partial Y_1}{\partial h} = a\left[(h\cos F - k\sin F)\frac{hk\beta^3}{1-\beta} - 1 + \frac{a}{r}\cos F(k\beta - \cos F)\right] \tag{A.6}$$

$$\frac{\partial Y_1}{\partial k} = a\left[(h\cos F - k\sin F)\left(\beta + \frac{k^2\beta^3}{(1-\beta)}\right) - \frac{a}{r}\sin F(\cos F - k\beta)\right] \tag{A.7}$$

The partial derivatives of the position vector with respect to the elements p and q are written after observing that X_1 and Y_1 are not functions of p and q. The unit vectors $\hat{f}$ and $\hat{g}$, however, are functions of p and q as shown in equations 2.35 and 2.36. Chobotov writes the partials of the position vector with respect to p and q as

follows.

$$\frac{\partial \boldsymbol{r}}{\partial p} = X_1 \frac{\partial \hat{f}}{\partial p} + Y_1 \frac{\partial \hat{g}}{\partial p} = \frac{2}{1+p^2+q^2}[q(Y_1\hat{f} - X_1\hat{g}) - X_1\hat{w}] \tag{A.8}$$

$$\frac{\partial \boldsymbol{r}}{\partial q} = X_1 \frac{\partial \hat{f}}{\partial q} + Y_1 \frac{\partial \hat{g}}{\partial q} = \frac{2}{1+p^2+q^2}[p(X_1\hat{g} - Y_1\hat{f}) + Y_1\hat{w}] \tag{A.9}$$

Finally, the partial derivative of the position vector with respect to the mean longitude can be written by utilizing the relationship between the mean longitude and the eccentric longitude.

$$\frac{\partial \boldsymbol{r}}{\partial \lambda} = \frac{\partial \boldsymbol{r}}{\partial F}\frac{\partial F}{\partial \lambda} = \frac{\dot{\boldsymbol{r}}}{n} \tag{A.10}$$

The last terms needed to derive the EOM are the Poisson brackets in terms of the equinoctial elements, which can be derived using the transformation presented in equation 2.65. The middle term in the equation, $[(a_\lambda, a_\mu)]$, represents the skew-symmetric matrix of Poisson brackets in terms of the classical orbital elements, and is defined as follows.

$$[(a_\lambda, a_\mu)] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -2\left(\frac{a}{\mu}\right)^{1/2} \\ & 0 & 0 & 0 & \frac{(1-e^2)^{1/2}}{e(\mu a)^{1/2}} & \frac{-(1-e^2)}{e(\mu a)^{1/2}} \\ & & 0 & \frac{1}{(\mu p)^{1/2}\sin i} & \frac{-\cos i}{(\mu p)^{1/2}\sin i} & 0 \\ & & & 0 & 0 & 0 \\ & & & & 0 & 0 \\ -Sym & & & & & 0 \end{bmatrix} \tag{A.11}$$

where $p = a(1-e^2)$. The other two matrices in equation 2.65 are comprised of the partial derivatives of the equinoctial elements with respect to the classical elements. Since the two matrices are transposes of each other only the first matrix is shown

below.

$$\left[\frac{\partial p_\alpha}{\partial a_\lambda}\right] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{h}{\sqrt{h^2+k^2}} & 0 & k & k & 0 \\ 0 & \frac{k}{\sqrt{h^2+k^2}} & 0 & -h & -h & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & \frac{p(1+p^2+q^2)}{2\sqrt{p^2+q^2}} & q & 0 & 0 \\ 0 & 0 & \frac{q(1+p^2+q^2)}{2\sqrt{p^2+q^2}} & -p & 0 & 0 \end{bmatrix} \tag{A.12}$$

After carrying out the matrix multiplications Chobotov presents the partials of the equinoctial elements with respect to velocity as the six by three matrix M.

$$M = \begin{bmatrix} \left(\frac{\partial a}{\partial \dot{\boldsymbol{r}}}\right)^T \\ \left(\frac{\partial h}{\partial \dot{\boldsymbol{r}}}\right)^T \\ \left(\frac{\partial k}{\partial \dot{\boldsymbol{r}}}\right)^T \\ \left(\frac{\partial p}{\partial \dot{\boldsymbol{r}}}\right)^T \\ \left(\frac{\partial q}{\partial \dot{\boldsymbol{r}}}\right)^T \\ \left(\frac{\partial \lambda}{\partial \dot{\boldsymbol{r}}}\right)^T \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \\ M_{41} & M_{42} & M_{43} \\ M_{51} & M_{52} & M_{53} \\ M_{61} & M_{62} & M_{63} \end{bmatrix} \tag{A.13}$$

where

$$M_{11} = \frac{2\dot{X}_1}{n^2 a} \tag{A.14}$$

$$M_{12} = \frac{2\dot{Y}_1}{n^2 a} \tag{A.15}$$

$$M_{13} = 0 \tag{A.16}$$

$$M_{21} = \frac{\sqrt{1-h^2-k^2}}{na^2}\left(\frac{\partial X_1}{\partial k} - \frac{h\beta}{n}\dot{X}_1\right) \tag{A.17}$$

$$M_{22} = \frac{\sqrt{1-h^2-k^2}}{na^2}\left(\frac{\partial Y_1}{\partial k} - \frac{h\beta}{n}\dot{Y}_1\right) \tag{A.18}$$

$$M_{23} = \frac{k(qY_1 - pX_1)}{na^2\sqrt{1-h^2-k^2}} \tag{A.19}$$

$$M_{31} = -\frac{\sqrt{1-h^2-k^2}}{na^2}\left(\frac{\partial X_1}{\partial h} + \frac{k\beta}{n}\dot{X}_1\right) \tag{A.20}$$

$$M_{32} = -\frac{\sqrt{1-h^2-k^2}}{na^2}\left(\frac{\partial Y_1}{\partial h} + \frac{k\beta}{n}\dot{Y}_1\right) \tag{A.21}$$

$$M_{33} = -\frac{h(qY_1 - pX_1)}{na^2\sqrt{1-h^2-k^2}} \tag{A.22}$$

$$M_{41} = 0 \tag{A.23}$$

$$M_{42} = 0 \tag{A.24}$$

$$M_{43} = \frac{(1+p^2+q^2)Y_1}{2na^2\sqrt{1-h^2-k^2}} \tag{A.25}$$

$$M_{51} = 0 \tag{A.26}$$

$$M_{52} = 0 \tag{A.27}$$

$$M_{53} = \frac{(1+p^2+q^2)X_1}{2na^2\sqrt{1-h^2-k^2}} \tag{A.28}$$

$$M_{61} = \frac{1}{na^2}\left[-2X_1 + \sqrt{1-h^2-k^2}\left(h\beta\frac{\partial X_1}{\partial h} + k\beta\frac{\partial X_1}{\partial k}\right)\right] \tag{A.29}$$

$$M_{62} = \frac{1}{na^2}\left[-2Y_1 + \sqrt{1-h^2-k^2}\left(h\beta\frac{\partial Y_1}{\partial h} + k\beta\frac{\partial Y_1}{\partial k}\right)\right] \tag{A.30}$$

$$M_{63} = \frac{qY_1 - pX_1}{na^2\sqrt{1-h^2-k^2}} \tag{A.31}$$

The componets of the M matrix can then be used to form the full set of EOM in terms of the equinoctial elements as shown in equations 2.66 through 2.71.

Appendix B. Optimal Control Algorithm in MATLAB

This Appendix includes the code that was written for the optimal control algorithm. The code is comprised of a main m-file that is used to run the algorithm, and includes two functions, defining the cost functional being mimized, and the differential equations governing the system dynamics. For the calculation of sidereal time, a series of functions are called that are not provided here. These scripts were taken from Vallado [11].

Main Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MODEL FOR CO-PLANAR MANEUVERS
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012

% OBJECTIVE: The objective of this code is to compute the minimum-time
% optimal trajectory for a satellite under constant low thrust.  The
% maneuvers considered here are co-planar, and the thrust vector is
% restricted to lie on the plane of the velocity vector.

% MANEUVER PROFILE: The satellite starts at a circular orbit with a
% given altitude, and will maneuver to achieve a given true anomaly, with
% the intent of overflying a ground target, at the minimum amount of time.
% The final altitude of the satellite is not constrained.  The magnitude
% of the thrust vector is assumed constant, which results in a constant
% acceleration applied to the satellite.  The direction of the acceleration
% vector is treated as the control variable.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all; clear all; clc;
format long;
```

```
% Declare global variables
global mu omega_earth gamma_g ar ah atheta solution a0 nu0 nuf A

% Define Constants
mu = 3.986e5; % km^3/s^2 - Earth's Gravitational constant
omega_earth = 7.2921e-5; % rad/s - Earth's rotation rate

%% PROPAGATE REFERENCE ORBIT
% NOTE: The reference orbit is the satellite's orbit with only two-body
% motion taken into consideration, i.e no maneuvering acceleration

% Enter Initial Conditions - User Input
a0 = 6878; % Initial semi-major axis, km
e0 = 0; % Initial eccentricity, unitless
i0 = 90*pi/180; % Initial inclination, rad
Omega0 = 0; % Initial RAAN, rad
w0 = 0; % initial argument of perigee ,rad
nu0 = 0; % initial true anomaly, rad

% Compute period of reference orbit
Pref = 2*pi*sqrt(a0^3/mu); % sec
% Define start and end times for orbit propagation as a function of orbital
% period
ti = 0; % Start time
tf = 15*Pref; % End time
t_step = 60; % step size, sec
t_vec = ti:t_step:tf; % time vector for numerical integration

% Setup Variables for Integration
```

```
% Enter the start date  and time to calculate appropriate Greenwich
% sidereal time (GST)
Yr = 2011; % Year
Mo = 6; % Month, 1-12
D = 14; % Day, 1-31
H = 16; % Universal Time, Hour 1-23
M = 0; % Universal Time, Minutes 0-59
S = 0; % Universal Time, Seconds 0-59.999
% NOTE: Functions "julianday", "wgs84data", and "gstime" are required
JD = julianday(Yr,Mo,D,H,M,S); % Calculates Julian Date
wgs84data % load global conversion factors
gamma_g = gstime(JD); % Clalculates GST at maneuver start

% Define intial state vector of COEs
x0 = [a0 e0 i0 Omega0 w0 nu0];

% Define Acceleration vector components
ar = 0; % radial acceleration, km/s^2
ah = 0; % normal acceleration, km/s^2
atheta = 0; % tangential acceleration, km/s^2

% Vehicle Parameters
m = 1000; % kg, Vehicle mass

% Numerical Integration using ode45
options = odeset('MaxStep',6);  %set ODE45 options
% NOTE: Function "LPE_Accel" required
[t_ref,x_ref]=ode45(@LPE_Accel,t_vec,x0,options);  %call ODE45
```

```
% Separate Variables
a_ref = x_ref(:,1); % Semi-major axis, km
e_ref = x_ref(:,2); % Eccentricity, unitless
i_ref = x_ref(:,3).*180/pi; % Inclination, deg
Omega_ref = x_ref(:,4)*180/pi; % RAAN, deg
w_ref = x_ref(:,5)*180/pi; % Argument of perigee, deg
nu_ref = x_ref(:,6)*180/pi; % True anomaly, deg
% These are the COEs for the reference orbit propagated for the desired
% amount of time
% SANITY CHECK: Since the reference orbit represents 2-body motion the COEs
% should be constant!

% Convert time to hours for plots
t_hr_ref = t_ref./(3600);

% From numerical intergration results calculate r and v in the ECF frame
% NOTE: Function "coe2rv" is required
[x, y, z, xdot, ydot, zdot] = ...
    coe2rv(a_ref, e_ref, i_ref, Omega_ref, w_ref, nu_ref, t_ref);
% Calculate latitude and longitude from ECF position
fi_ref = atan2(y,x)*180/pi; % longitude, deg
lamda_ref = asin(z./sqrt(x.^2+y.^2+z.^2))*180/pi; % latitude, deg
% These values can be used to plot the ground track of the reference orbit

%% OPTIMIZATION ALGORITHM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The Optimization package used in this model is GPOPS 4.1
% Software is required to run this portion of the code
% GPOPS is run twice: the first run uses a coarse grid and a larger
```

```
% tolerance to get an intial solution; the second run uses a finer grid and
% smaller tolerance and uses the first run's result as a guess in order to
% get a refined solution.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% GPOPS First Run
% Initialize GPOPS for first run
run('I:\THESIS\MATLAB Model\GPOPS41\gpops\gpopsSetup.m')
% The path input in the command needs to be valid for the computer that is
% being used. For example if I was running this on my home computer the
% command would look like:
% run('C:\Users\Costas\Documents\Costas AFIT Stuff\Thesis\MATLAB Model\...
% GPOPS41\gpops\gpopsSetup.m')

% User-defined constants
nuf = 14*2*pi + 180*pi/180; % final desired true anomaly
T = 1; % Constant thrust, Newtons
A = T/m*10^(-3); % constant acceleration magnitude, km/s^2

% Initial and final conditions
a0 = a0; % initial orbit radius
% Define final altitude if it is to be constrained
% af = a0;
nu0 = nu0; % initial true anomaly
% The dynamics equations have been simplified and only semi-major axis and
% true anomaly are considered

% Limits during maneuver
amin = 6578; % km, cannot go below this altitude
```

```
amax = 100000; % km, cannot go above this altitude
numin = 0; % rad, negative true anomaly not valid
numax = 15*2*pi; % rad, cannot go above this value - since the desired
% final true anomaly is 20 revolutions (40*pi) ahead the max true anomaly
% will not be more than 21 full revolutions

% Control and time boundaries
umin =  -pi/2; % minimum control angle
umax =  pi/2; % maximum control angle
t0min = 0; % minimum initial time
t0max = 0; % maximum initial time
tfmin = 0; % minimum final time
tfmax = 15*Pref; % maximum final time

% GPOPS Setup
% Phase 1 Information
iphase = 1;
limits(iphase).intervals = 1;
limits(iphase).nodesperint = 10;
limits(iphase).time.min = [t0min tfmin];
limits(iphase).time.max = [t0max tfmax];
% LIMITS ON STATE AND CONTROL VALUES THROUGHOUT TRAJECTORY
% to constrain final altitude use:
% limits(iphase).state.min(1,:) = [a0 amin af];
% limits(iphase).state.max(1,:) = [a0 amax af];
% to leave final altitude unconstrained use:
limits(iphase).state.min(1,:) = [a0 amin amin];
limits(iphase).state.max(1,:) = [a0 amax amax];
limits(iphase).state.min(2,:) = [nu0 numin nuf];
```

```
limits(iphase).state.max(2,:) = [nu0 numax nuf];
limits(iphase).control.min = umin;
limits(iphase).control.max = umax;
% LIMITS ON PARAMETERS, PATH, AND EVENT CONSTRAINTS
limits(iphase).parameter.min = []; % None
limits(iphase).parameter.max = []; % None
limits(iphase).path.min = []; % None
limits(iphase).path.max = []; % None
limits(iphase).event.min = []; % None
limits(iphase).event.max = []; % None
% GUESS SOLUTION
guess(iphase).time = [t0min; tfmax];
% Semi-major axis guess for constrained altitude:
% guess(iphase).state(:,1) = [a0; af];
% Semi-major axis guess for unconstrained altitude:
guess(iphase).state(:,1) = [a0; amin];
guess(iphase).state(:,2) = [nu0; nuf];
% Control guess for constrained altitude:
% guess(iphase).control = [-pi/2; pi/2];
% Control guess for unconstrained altitude:
guess(iphase).control = [-pi/2; -pi/2];
guess(iphase).parameter = []; % No parameters in Phase 1

% NOTE: Functions "phasingmaneuverCost" and "phasingmaneuverDae" required
linkages = [];
setup.name  = 'Phasing-Maneuver-Problem';
setup.method = 'gauss';
setup.funcs.cost = 'phasingmaneuverCost';
setup.funcs.dae = 'phasingmaneuverDae';
```

```
setup.limits = limits;
setup.guess = guess;
setup.linkages = linkages;
setup.derivatives = 'automatic';
setup.direction = 'increasing';
setup.autoscale = 'off';
setup.solver = 'snopt';
setup.mesh.grid = 'hp';
setup.mesh.tolerance = 1e-2;
setup.mesh.iteration = 5;
setup.mesh.on = 'yes';
setup.mesh.guess = 'yes';
setup.mesh.nodelimit = 200;

output = gpops(setup);
solution = output.solution;
solution.control = unwrap(solution.control);
save initial_guess

%% GPOPS Second Run
% Initialize GPOPS for second run
% run('I:\THESIS\MATLAB Model\GPOPS-v31\GPOPS 3.1\gpopsinitialize.m')
% The path input in the command needs to be valid for the computer that is
% being used. For example if I was running this on my home computer the
% command would look like:
% run('C:\Users\Costas\Documents\Costas AFIT Stuff\Thesis\MATLAB Model\...
% GPOPS-v31\GPOPS 3.1\gpopsinitialize.m')

% User-defined constants
```

```
nuf = 14*2*pi + 180*pi/180; % final desired true anomaly
T = 1; % Constant thrust, Newtons
A = T/m*10^(-3); % constant acceleration magnitude, km/s^2

% Initial and final conditions
a0 = a0; % initial orbit radius
% Define final altitude if it is to be constrained
% af = a0;
nu0 = nu0; % initial true anomaly
% The dynamics equations have been simplified and only semi-major axis and
% true anomaly are considered

% Limits during maneuver
amin = 6578; % km, cannot go below this altitude
amax = 100000; % km, cannot go above this altitude
numin = 0; % rad, negative true anomaly not valid
numax = 15*2*pi; % rad, cannot go above this value - since the desired
% final true anomaly is 20 revolutions (40*pi) ahead the max true anomaly
% will not be more than 21 full revolutions

% Control and time boundaries
umin =  -pi; % minimum control angle
umax =  pi; % maximum control angle
t0min = 0; % minimum initial time
t0max = 0; % maximum initial time
tfmin = 0; % minimum final time
tfmax = 15*Pref; % maximum final time

% Define guess based on previous solution
```

```
a_guess = solution.state(:,1);
nu_guess = solution.state(:,2);
u_guess = solution.control;
t_guess = solution.time;
clear solution

%GPOPS Setup
% Phase 1 Information
iphase = 1;
limits(iphase).intervals = 1;
limits(iphase).nodesperint = 10;
limits(iphase).time.min = [t0min tfmin];
limits(iphase).time.max = [t0max tfmax];
% LIMITS ON STATE AND CONTROL VALUES THROUGHOUT TRAJECTORY
% to constrain final altitude use:
% limits(iphase).state.min(1,:) = [a0 amin af];
% limits(iphase).state.max(1,:) = [a0 amax af];
% to leave final altitude unconstrained use:
limits(iphase).state.min(1,:) = [a0 amin amin];
limits(iphase).state.max(1,:) = [a0 amax amax];
limits(iphase).state.min(2,:) = [nu0 numin nuf];
limits(iphase).state.max(2,:) = [nu0 numax nuf];
limits(iphase).control.min = umin;
limits(iphase).control.max = umax;
% LIMITS ON PARAMETERS, PATH, AND EVENT CONSTRAINTS
limits(iphase).parameter.min = []; % None
limits(iphase).parameter.max = []; % None
limits(iphase).path.min = []; % None
limits(iphase).path.max = []; % None
```

```
limits(iphase).event.min = []; % None
limits(iphase).event.max = []; % None
% GUESS SOLUTION
clear guess
guess(iphase).time = t_guess;
guess(iphase).state(:,1) = a_guess;
guess(iphase).state(:,2) = nu_guess;
guess(iphase).control = u_guess;
guess(iphase).parameter = []; % No parameters in Phase 1

linkages = [];
setup.name  = 'Phasing-Maneuver-Problem';
setup.method = 'gauss';
setup.funcs.cost = 'phasingmaneuverCost';
setup.funcs.dae = 'phasingmaneuverDae';
setup.limits = limits;
setup.guess = guess;
setup.linkages = linkages;
setup.derivatives = 'automatic';
setup.direction = 'increasing';
setup.autoscale = 'off';
setup.solver = 'snopt';
setup.mesh.grid = 'hp';
setup.mesh.tolerance = 1e-3;
setup.mesh.iteration = 25;
setup.mesh.on = 'yes';
setup.mesh.guess = 'yes';
setup.mesh.nodelimit = 200;
```

```
output = gpops(setup);
solution = output.solution;
solution.control = unwrap(solution.control);

% Define variables
t_soln = solution.time; % time, s
t_min_soln = t_soln/60; % time, min
t_hr_soln = t_soln/3600; % time, hr
a_soln = solution.state(:,1); % semi-major axis, km
nu_soln = solution.state(:,2); % true anomaly, rad
nu_deg = nu_soln.*180/pi; % true anomaly, deg
th = solution.control; % control angle, rad
th_deg = th.*180/pi; % control angle, deg

%% PROPAGATE ORBIT FOR MANEUVERING SATELLITE
% Define time vector for integration
t_vec_man = t_soln(1):60:t_soln(end);
% Numerical Integration using ode45
options = odeset('MaxStep',6);  %set ODE45 options
% NOTE: Function "LPE_Accel_maneuver" required
[t_man,x_man]=ode45(@LPE_Accel_maneuver,t_vec_man,x0,options);  %call ODE45

% Separate Variables
a_man = x_man(:,1); % semi-major axis, km
e_man = x_man(:,2); % eccentricity
i_man = x_man(:,3).*180/pi; % inclination, deg
Omega_man = x_man(:,4)*180/pi; % RAAN, deg
w_man = x_man(:,5)*180/pi; % Argument of perigee, deg
nu_man = x_man(:,6)*180/pi; % True anomaly, deg
```

```
% Convert time to hours for plots
t_min_man = t_man./(60);
t_hr_man = t_man./3600;

% From numerical intergration results calculate r and v in the ECF frame
% NOTE: Function "coe2rv" is required
[x, y, z, xdot, ydot, zdot] =...
    coe2rv(a_man, e_man, i_man, Omega_man, w_man, nu_man, t_man);
% Calculate latitude and longitude from ECF position
fi_man = atan2(y,x)*180/pi; % longitude, deg
lamda_man = asin(z./sqrt(x.^2+y.^2+z.^2))*180/pi; % latitude, deg
% These values can be used to plot the ground track of the maneuvering orbit

%% PLOT OPTIMAL SOLUTION
figure % Open new figure
subplot(221)
plot(t_hr_soln,a_soln)%,'Linewidth',1.5)
xlabel('Time, hr')
ylabel('Semi-major axis, km')
grid on
subplot(222)
plot(t_hr_soln,nu_deg)%,'Linewidth',1.5)
xlabel('Time, hr')
ylabel('True Anomaly, deg')
grid on
subplot(223)
plot(t_hr_soln,th_deg)%,'Linewidth',1.5)
xlabel('Time, hr')
```

```
ylabel('Control Angle, deg')
grid on
% Plot Eccentricity
subplot(224)
plot(t_hr_man,e_man)%,'Linewidth',1.5)
xlabel('Time, hr')
ylabel('Eccentricity')
grid on
%suptitle ('Optimal Solution')

%% PLOT GROUND TRACK
index = find(t_ref >= 14*Pref);
% plot ground track
landareas = shaperead('landareas.shp','UseGeoCoords',true);
figure
geoshow(landareas)
hold on
plot(fi_ref(index(1):end),lamda_ref(index(1):end),'.','MarkerSize',5)
% plot(fi_man(index(1):end),lamda_man(index(1):end),'r.','MarkerSize',5)
% plot(fi_tgt,lamda_tgt,'ro','MarkerSize',10)
plot(fi_man(end),lamda_man(end),'^','MarkerFaceColor','k','MarkerSize',10)
ind = find(t_ref == t_man(end));
plot(fi_ref(ind),lamda_ref(ind),'o','MarkerFaceColor','k','MarkerSize',10)
grid on
xlabel('Longitude, deg','fontsize',14)
ylabel('Latitude, deg','fontsize',14)
legend('Reference Ground Track','Position of Maneuvering Satellite at t_f',...
    'Position of Reference Satellite at t_f')
hold off
```

Cost Functional

```
function [Mayer,Lagrange]=phasingmaneuverCost(sol)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012
%
% This function file is the cost function that is to be extremized using
% GPOPS. The function defines a Mayer cost and a Lagrange cost separately
% and GPOPS combines the two to form the Bolza cost function.  The input to
% the function file is the optimal solution from GPOPS and the outputs are
% the Mayer cost and the Lagrange cost.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define variables for time and final time
tf = sol.terminal.time;
t  = sol.time;
% Define  Mayer cost
Mayer = tf;
% Define weighting factor for control usage
alpha = 0e-4; % Set to zero if no penalty on control usage is needed
% Define Lagrange Cost
Lagrange = alpha*(sol.control.*sol.control);
```

Differential Algebraic Equations

```
function daeout = phasingmaneuverDae(sol)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012
%
```

```
% This function file defines the simplified first order differential
% equations governing the dynamics of the system.  GPOPS uses these
% equations to enforce constraints on the optimal solution.  The input to
% this function file is the optimal solution, and the output is a matrix
% containing the state derivatives.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% declare global variables
global a0 mu nu0 nuf A

% Define state and control variables
t = sol.time;
s = sol.state;
u = sol.control;
a = s(:,1); % First state - semi-major axis
nu = s(:,2); % Second state - true anomaly

% Define first order ODEs governing state dynamics
adot = 2/sqrt(mu).*a.^(3/2).*A.*sin(u);
nudot = sqrt(mu./(a.^3));

% Form matrix output
daeout = [adot nudot];
```

Appendix C. Feedback Control Algorithm in MATLAB

This Appendix includes the code that was written for the feedback control algorithm. The code is comprised of a main m-file that is used to run the algorithm, and two function files. The first function file is the ODE45 script used to propagate the maneuver trajectory and includes the computation of the control input. The second function is used to solve the analytic expression for Δt, and calculate the required maneuvering time and drifting time. For the calculation of sidereal time, a series of functions are called that are not provided here. These scripts were taken from Vallado [11].

Main Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MODEL FOR LYAPUNOV FEEDBACK CONTROLLER
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012

% OBJECTIVE: The objective of this code is to compute a low-thrust orbit
% transfer trajectory using a nonlinear Lyapunov feedback controller.  The
% spacecraft is required to achieve 30 seconds of Delta-t, for a given
% amount of lead time.

% MANEUVER PROFILE: The satellite starts at a near circular orbit with a
% given altitude, and will maneuver to achieve a given set of orbital
% elements.  The Lyapunov controller computes a control unit vector to
% achieve the desired changes in orbital elements. Magnitude of the
% acceleration control vector is assumed constant (no throttling).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all; clear all; clc;
```

```
format long;

% Declare global variables
global mu Re omega_earth gamma_g m0 T A g0 Isp...
       a_des e_des i_des Omega_des w_des nu_des...
       counter u_hist t_hist u_norm_hist x_hist DeltaV DeltaV_req t_start...
       thruster_on thruster_off

% Define constants
mu = 3.986e5; % km^3/s^2
Re = 6378.137; % Equatorial Earth Radius, km
omega_earth = 7.2921e-5; % Earth's rotation rate, rad/s
counter = 0; % counter variable used to keep track of values computed
             % throughout numerical integration
% Vehicle Parameters
m0 = 1000; % kg, Vehicle mass
T = 1; % Constant thrust, Newtons
g0 = 9.80665; % gravitational constant at sea level, m/s^2
Isp = 1500; % Engine specific Impulse, s

% Duty cycle parameters - COMMENT OUT IF NOT NEEDED
thruster_on = 10; % duration of pulses, sec
thruster_off = 120; % duration between pulsed, sec
ratio = thruster_on/(thruster_on+thruster_off);
A_approx = ratio*T/m0*10^(-3); % approximate constant acceleration
                       % magnitude used to compute required DeltaV, km/s^2

%% DETERMINE TARGET ORBIT PARAMETERS
% Enter the date to calculate appropriate Greenwich sidereal time at epoch
```

```
Yr = 2011; % Year
Mo = 6; % Month, 1-12
D = 14; % Day, 1-31
H = 16; % Universal Time, Hour 1-23
M = 0; % Universal Time, Minutes 0-59
S = 0; % Universal Time, Seconds 0-59.999
JD = julianday(Yr,Mo,D,H,M,S); % Calculates Julian Date
wgs84data % load global conversion factors
gamma_g = gstime(JD); % Clalculates Greenwich sidereal time at start

%% INITIAL CONDITIONS
a0 = 7378; % Initial semi-major axis, km
e0 = 0; % Initial eccentricity
i0 = 45*pi/180; % Initial inclination, rad
Omega0 = 0; % Initial RAAN, rad
w0 = 0; % initial argument of perigee ,rad
nu0 = 0; % initial mean anomaly, rad
% Compute period of reference orbit
Pref = 2*pi*sqrt(a0^3/mu); % sec

% Define desired timing parameters
delT_des = 30; % desired amount of delta-t, sec
t_avail = 24*3600; % total available amount of time (lead time), sec
t_start = 0*3600; % initial time before spacecraft starts thrusting, sec

% Define integration timing
ti = 0; % Start time
tf = t_avail; % End time, sec
t_step = 60; % step size, sec
```

```
t_vec = ti:t_step:tf; % time vector

%% PROPAGATE REFERENCE ORBIT
% Define intial state vector and integration timing
x0_ref = [a0 e0 i0 Omega0 w0 nu0];

% Numerical Integration using ode45
options = odeset('MaxStep',6);  %set ODE45 options
% NOTE: Function "LPE_Accel" required
[t_ref,x_ref]=ode45(@LPE_Accel,t_vec,x0_ref,options);  %call ODE45

% Separate Variables
a_ref = x_ref(:,1); % Semi-major axis, km
e_ref = x_ref(:,2); % Eccentricity, unitless
i_ref = x_ref(:,3).*180/pi; % Inclination, deg
Omega_ref = x_ref(:,4)*180/pi; % RAAN, deg
w_ref = x_ref(:,5)*180/pi; % Argument of perigee, deg
nu_ref = x_ref(:,6)*180/pi; % True anomaly, deg
% These are the COEs for the reference orbit propagated for the desired
% amount of time
% SANITY CHECK: Since the reference orbit represents 2-body motion the COEs
% should be constant!

% Convert time to hours for plots
t_hr_ref = t_ref./(3600);

% From numerical intergration results calculate r and v in the ECF frame
% NOTE: Function "coe2rv" is required
[x, y, z, xdot, ydot, zdot] = ...
```

```
    coe2rv(a_ref, e_ref, i_ref, Omega_ref, w_ref, nu_ref, t_ref);
% Calculate latitude and longitude from ECF position
sat_long_ref = atan2(y,x)*180/pi; % longitude, deg
sat_lat_ref = asin(z./sqrt(x.^2+y.^2+z.^2))*180/pi; % latitude, deg

%% COMPUTE REQUIRED DELTA-V AND CONTROL INPUT
% Compute true anomaly after intial coasting period
nu_start = nu0 + (t_start)*sqrt(mu/(a0^3));
% Compute desired parameters
[nu2,a_final,DeltaV_req,t_man,t_coast] = ...
    thrust_coast_combo(a0, nu_start, t_avail-t_start, delT_des, A_approx);
% Check to see if unique solution was found
if isempty(t_man)
    disp('ERROR: The Delta-t requested is not achievable in the time available')
else
% Desired Final Parameters
    a_des = a_final; % desired semi-major axis
    e_des = e0; % desired eccentricity, no change
    i_des = i0; % desired inclination, no change
    Omega_des = Omega0; % desired RAAN, no change
    w_des = 0; % desired argument of perigee, arbitrary
    nu_des = nu2; % desired true anomaly

% Initial State Vector
    x0 = [a0 e0 i0 Omega0 w0 nu0 m0];

% PROPAGATE ORBIT
    options = odeset('MaxStep',6);  %set ODE45 options
    [t,x]=ode45(@LPE_Accel_Lyap,t_vec,x0,options); %call ODE45
```

```
% Separate Variables
    a = x(:,1); % Semi-major axis, km
    e = x(:,2); % Eccentricity
    i = x(:,3).*180/pi; % Inclination, deg
    Omega = x(:,4)*180/pi; % RAAN, deg
    w = x(:,5)*180/pi; % Argument of Perigee, deg
    nu = x(:,6)*180/pi; % True anomaly, deg

% Convert time to hours for plots
    t_hr = t./(3600); % time vector associated with states
    t_hist_plt = t_hist./(3600); % time vector associated with control

% Compute control angles from control vector
    beta = asin(u_hist(3,:)./A)*180/pi; % Out-of-plane angle, deg
    alpha = atan2(u_hist(2,:)./A, u_hist(1,:)./A)*180/pi; % In-plane angle

% Calculate r and v in ECI and ECF frames
    [x_ecf, y_ecf, z_ecf, xdot, ydot, zdot, x_eci, y_eci, z_eci] = ...
        coe2rv(a, e, i, Omega, w, nu, t);
% Calculate latitude and longitude from ECF position
    sat_long = atan2(y_ecf,x_ecf)*180/pi; % longitude, deg
    sat_lat = asin(z_ecf./sqrt(x_ecf.^2+y_ecf.^2+z_ecf.^2))*180/pi;
                                                    % latitude, deg

% PLOT SOLUTIONS
    figure % plot numerically integrated states
    subplot(2,2,1); plot(t_hr,a); grid on
    xlabel('Time, hrs')
```

```
    ylabel('Semi-major Axis, km')
    subplot(2,2,2); plot(t_hr,e); grid on
    xlabel('Time, hrs')
    ylabel('Eccentricity')
    subplot(2,2,3); plot(t_hr,nu); grid on
    xlabel('Time, hrs')
    ylabel('True Anomaly, deg')
    subplot(2,2,4); plot(t_hist_plt,alpha); grid on
    xlabel('Time, hrs')
    ylabel('In-plane Angle, deg')

% plot ground track
% pick values for last two orbits
    num_orbits = round(t_avail/Pref) - 2;
    index = find(t_ref >= num_orbits*Pref);
    landareas = shaperead('landareas.shp','UseGeoCoords',true);
    figure
    geoshow(landareas)
    hold on
    plot(sat_long_ref(index(1):end),sat_lat_ref(index(1):end),'.',...
        'MarkerSize',5)
    plot(sat_long(end),sat_lat(end),'^','MarkerFaceColor','k',...
        'MarkerSize',10)
    plot(sat_long_ref(end),sat_lat_ref(end),'o','MarkerFaceColor','k',...
        'MarkerSize',10)
    grid on
    xlabel('Longitude, deg')
    ylabel('Latitude, deg')
    title('Satellite Ground Track')
```

```
    legend('Reference Ground Track','Satellite Position at t_f',...
        'Reference Position at t_f')
    hold off
end
%% DISPLAY RESULTS
if isempty(t_man)
    % Delta t undefined if no solution is found
    Delta_t = [];
else
    % Compute Delta t achieved at the end of the maneuver
    Delta_t = nu2*sqrt(a0^3/mu) - t_avail;
end
fprintf('The achieved Delta-t is %f seconds\n',Delta_t)
fprintf('The expended Delta-V is %f km/s\n',DeltaV)
fprintf('The final semi-major axis is %f km\n',a_final)
```

ODE45 Function File

```
function[xdot]=LPE_Accel_Lyap(t,x)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012
%
% This function file defines the full set of first order differential
% equations governing the dynamics of the orbit in terms of orbital
% elements. A Lyapunov function is defined and used to compute a control
% vector in order to reach a desired state
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Declare global variables
```

```
global mu Re gamma_g m0 T A g0 Isp...
       a_des e_des i_des Omega_des w_des nu_des...
       counter u_hist t_hist u_norm_hist x_hist DeltaV DeltaV_req t_start...
       thruster_on thruster_off
% Update counter
counter = counter + 1;

% Set up variables
a = x(1); e = x(2); i = x(3); Omega = x(4); w = x(5); nu = x(6); m = x(7);

% Define orbital parameters (to be recalculated every iteration)
h = sqrt(mu*a*(1-e^2)); % angular momentum
p = a*(1-e^2); % semilatus rectum
theta = w+nu;
r = p/(1+e*cos(nu)); % radius
b = a*sqrt(1-e^2); % semi-minor axis

% Define controller gains
Ka = 1; % Gain associated with semi-major axis
Ke = 0; % Gain associated with eccentricity
Ki = 0; % Gain associated with inclination
KOmega = 0; % Gain associated with RAAN
Komega = 0; % Gain associated with argument of perigee
Knu = 0;
% Check if desired orbit plane is achieved
if i == i_des && Omega == Omega_des
    delta_nu = nu-nu_des; % Phase difference
    % Implement Mean Motion Control
    K = 10; % Gain associates with mean motion control
```

```
    a_star = (-K/sqrt(mu)*delta_nu+1/(a^3/2))^(-2/3);
else
   a_star = a_des;
end
% Form Lie Derivative of Lyapunov function
V = [Ka*(a-a_star)/(Re^2), Ke*(e-e_des), Ki*(i-i_des),...
    KOmega*(Omega-Omega_des), Komega*(w-w_des), Knu*(nu-nu_des)];

% Form N matrix - last two rows set to zero to avoid singularities
N = [2*a^2/h*(e*sin(nu)), 2*a^2/h*p/r,                                  0;
    1/h*p*sin(nu),        1/h*((p+r)*cos(nu)+r*e),                      0;
    0,                    0,                                r*cos(theta)/h;
    0,                    0,                     r*sin(theta)/(h*sin(i));
    0,                    0,                                           0;
    0,                    0,                                          0];

% Compute control unit vector
u_norm = sqrt(V*N*N'*V'); % compute norm to make unit vector

% Compute Acceleration magnitude - accounting for change in mass
A(counter) = T/m*10^(-3); % km/s^2

% Compute the Delta-V added thus far
DeltaV = g0*Isp*log(m0/m)/1000; % km/s

% % for continuous thrusting:  (COMMENT OUT IF NOT NEEDED)
% % Check if norm = 0 to avoid dividing by zero
% if u_norm == 0
%     u = [0;0;0];
```

```
% % Check if DeltaV has reached the reuired value
% elseif DeltaV > DeltaV_req
%     u = [0;0;0];
% % Check if time is past the required initial coasting period
% elseif t < t_start
%     u = [0;0;0];
% else
%     u = -A(counter)/u_norm*(N'*V');
% end

% for thrust-coast cycle:   (COMMENT OUT IF NOT NEEDED)
% Check if norm = 0 to avoid dividing by zero
if u_norm == 0
   u = [0;0;0];
% Check if DeltaV has reached the reuired value
elseif DeltaV > DeltaV_req
    u = [0;0;0];
% Check if time is past the required initial coasting period
elseif t < t_start
    u = [0;0;0];
else % Implement thrust-coast duty cycle
    % always apply thrust at the first iteration
    if u_norm > 0 && counter == 1
        u = -A(counter)/u_norm*(N'*V');
    % determine if it's past the first iteration
    elseif u_norm > 0 && counter > 1
        ind = find(u_norm_hist); % return indices of non-zero values
        if isempty(ind) % all entries are zero
            u = -A(counter)/u_norm*(N'*V');
```

```
elseif length(ind) == length(u_norm_hist) % all entries non-zero
   thrust_time = t_hist(ind(end))-t_hist(ind(1));
   % determine if thrust has been applied for 10 seconds or more
   if thrust_time >= thruster_on
       u = [0;0;0];
   else
       u = -A(counter)/u_norm*(N'*V');
   end
else
 % there are zero and non-zero entries in the control norm vector
 % return indices of zero values
    ind_zeros = find(u_norm_hist == 0);
    % determine where the last non-zero value is
    if ind_zeros(end) < ind(end)
        thrust_time = t_hist(ind(end))-t_hist(ind_zeros(end)+1);
     % determine if thrust has been applied for 10 seconds or more
        if thrust_time >= thruster_on
            u = [0;0;0];
        else
            u = -A(counter)/u_norm*(N'*V');
        end
    else
    k = 1; % initialize counter variable for loop
    while k < ind_zeros(end)
   % determine if the value before the last zero entry is non-zero
        if u_norm_hist(ind_zeros(end)-k) == 0
             k = k+1; % update counter
        else
            coast_time = t_hist(ind_zeros(end))-...
```

```
                        t_hist(ind_zeros(end)-k);
                    % detrmine if there are non-zero entries AFTER
                    % the last zero entry
                    if ind(end) > ind_zeros(end)
                        thrust_time = t_hist(ind(end))-...
                            t_hist(ind_zeros(end)+1);
                    else
                        thrust_time = 0;
                    end
                    % determine if coasting time is more than 2 minutes
                    % and thrust has been applied for more than 10 seconds
                    if coast_time > thruster_off && thrust_time < thruster_on
                        u = -A(counter)/u_norm*(N'*V');
                    else
                        u = [0;0;0];
                    end
                    break % break out of loop
                end
            end
            end
        end
    end
end

u_hist(:,counter) = u; % save control input throughout integration
x_hist(counter,:) = x; % save state history
u_norm_hist(counter) = norm(u); % save norm of control input
t_hist(counter) = t; % save time variable throughout integration
```

```
% Define first order ODEs governing state dynamics
adot = 2*a^2/h*(e*sin(nu)*u(1) + p/r*u(2)); % semi-major axis change
edot = 1/h*(p*sin(nu)*u(1) + ((p+r)*cos(nu)+r*e)*u(2)); %eccentricity change
idot = r*cos(theta)/h*u(3); % inclination change
Omegadot = r*sin(theta)/(h*sin(i))*u(3); % RAAN change
wdot = 0; % remove singularity - ignore argument of perigee
nudot = sqrt(mu/(a^3)); % remove singularity - assume circular orbit
                        % throughout trajectory
if u == [0;0;0];
    mdot = 0; % if no control input then mass doesn't change
else
    mdot = -T/(g0*Isp);
end

% Form matrix output
xdot = [adot edot idot Omegadot wdot nudot mdot]';
% end function file
```

Thrust-drift Function File

```
function [nu2,a_final,DeltaV_req,t_man,t_coast] = ...
    thrust_coast_combo(a0, nu0, t_avail, delT_des, A)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012
%
% This function file computes the required maneuvering and drifting time
% based on the amount of time available and the value of Delta-t desired.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global mu t_start
```

```
% from t_avail calculate nu2
nu2 = nu0 + (t_avail+delT_des)*sqrt(mu/(a0^3));
% define symbolic variable
syms x
% Define algebraic equation for Delta-t
eq = -delT_des + (nu2-nu0)*sqrt(a0^3/mu)-sqrt(mu)/A*((a0^(-2)+...
    4/mu*A*(x-nu0))^(1/4)-a0^(-1/2))-(nu2-x)*sqrt((a0^(-2)+...
    4/mu*A*(x-nu0))^(-3/2)/mu);
% Find solution
sol = solve(eq,x);
% Convert solution to type double from symbolic
nu1 = double(sol);
% Compute final semimajor axis reached
a_final = (a0^(-2)+4/mu*A*(nu1-nu0))^(-1/2);
% COmpute required maneuvering time and DeltaV
t_man = sqrt(mu)/A*((a0^(-2)+4/mu*A*(nu1-nu0))^(1/4)-a0^(-1/2));
DeltaV_req = A*t_man;
% Compute required drifting time
t_coast = (nu2-nu1)*sqrt((a0^(-2)+4/mu*A*(nu1-nu0))^(-3/2)/mu);
% end function file
```

This Appendix includes other functions that were written during the development of the algorithms.

ODE45 Function File for Reference Spacecraft

This function file propagates the equations of motion for the reference spacecraft.

```
function[xdot]=LPE_Accel(t,x)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012
%
% This function file defines the full set of first order differential
% equations governing the dynamics of the orbit in terms of orbital
% elements. These equations are used to propagate the reference orbit ahead
% in time.  The input to this function is time and state solutions computed
% by ode45, and the output is the state derivatives.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Declare global variables
global mu omega_earth gamma_g

% Set up variables
a = x(1); e = x(2); i = x(3); Omega = x(4); w = x(5); nu = x(6);

% Set acceleration to zero for reference case
ar = 0; atheta = 0; ah = 0;

% Define orbital parameters (to be recalculated every iteration)
h = sqrt(mu*a*(1-e^2)); % angular momentum
```

```
p = a*(1-e^2); % semilatus rectum
theta = w+nu;
r = p/(1+e*cos(nu)); % radius
b = a*sqrt(1-e^2); % semi-minor axis

% Define first order ODEs governing state dynamics
adot = 2*a^2/h*(e*sin(nu)*ar + p/r*atheta); % semi-major axis change
edot = 1/h*(p*sin(nu)*ar + ((p+r)*cos(nu)+r*e)*atheta); %eccentricity change
idot = r*cos(theta)/h*ah; % inclination change
Omegadot = r*sin(theta)/(h*sin(i))*ah; % RAAN change
% To avoid singularities set wdot to zero, and nu dot to the mean motion
wdot = 0; %1/(h*e)*(-p*cos(nu)*ar + (p+r)*sin(nu)*sin(nu)*atheta) - ...
        %r*sin(theta)*cos(i)/(h*sin(i))*ah; % arg of perigee change
nudot = h/(r^2); %h/(r^2)+1/(h*e)*(p*cos(nu)*ar - (p+r)*sin(nu)*atheta);
                % true anomaly change

% Form matrix output
xdot = [adot edot idot Omegadot wdot nudot]';

% end function file
```

ODE45 Function File for Maneuvering Spacecraft

This function file is used to propagate the full set of equations of motion using the optimal solution computed in the optimization algorithm. The function includes code that finds the appropriate value of the control angle throughout the integration, by fitting a cubic spline to the optimal solution.

```
function[xdot]=LPE_Accel_maneuver(t,x)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 9 Jan 2012
%
% This function file defines the full set of first order differential
% equations governing the dynamics of the orbit in terms of orbital
% elements. These equations are used to propagate the maneuver orbit ahead
% in time.  The input to this function is time and state solutions computed
% by ode45, and the output is the state derivatives.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Declare global variables
global mu omega_earth gamma_g solution A g0 Isp T

% Set up variables
a = x(1); e = x(2); i = x(3); Omega = x(4); w = x(5); nu = x(6);

% Define orbital parameters (to be recalculated every iteration)
h = sqrt(mu*a*(1-e^2)); % angular momentum
p = a*(1-e^2); % semilatus rectum
theta = w+nu;
r = p/(1+e*cos(nu)); % radius
b = a*sqrt(1-e^2); % semi-minor axis

% Optimal control from GPOPS solution
tt = solution.time; % optimal time solution
u = solution.control; % optimal control angle
% fit cubic spline on optimal control curve
u_spline = spline(tt,u); % returns polynomial coefficients
% evaluate cubic spline at time t
```

```
u_opt = ppval(u_spline,t);
% define acceleration components
ar_opt = A.*cos(u_opt);
atheta_opt = A.*sin(u_opt);
ah_opt = 0;

% Define first order ODEs governing state dynamics
adot = 2*a^2/h*(e*sin(nu)*ar_opt + p/r*atheta_opt); % semi-major axis change
edot = 1/h*(p*sin(nu)*ar_opt + ((p+r)*cos(nu)+r*e)*atheta_opt);
        %eccentricity change
idot = r*cos(theta)/h*ah_opt; % inclination change
Omegadot = r*sin(theta)/(h*sin(i))*ah_opt; % RAAN change
% To avoid singularities set wdot = 0 and nudot to the mean motion
wdot = 0; %1/(h*e)*(-p*cos(nu)*ar_opt + (p+r)*sin(nu)*sin(nu)*atheta_opt) - ...
        % r*sin(theta)*cos(i)/(h*sin(i))*ah_opt; % arg of perigee change
nudot = sqrt(mu/(a^3));
%h/(r^2)+1/(h*e)*(p*cos(nu)*ar_opt - (p+r)*sin(nu)*atheta_opt);
% true anomaly change

% Form matrix output
xdot = [adot edot idot Omegadot wdot nudot]';

% end function file
```

Conversion file from orbital elements to position/velocity

This function file is a conversion between the classical orbital elements and the position and velocity vectors, using the standard two body equations. This function is used to calculate the latitude and longitude of the spacecraft, used to plot the ground track.

```
function [x_ecf, y_ecf, z_ecf, xdot, ydot, zdot, x_eci, y_eci, z_eci] = ...
    coe2rv(a, e, i, Omega, w, nu, t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by: Capt Costas Zagaris, AFIT Master's Student
% Last Edited on: 5 Oct 2011
%
% This function file defines the position and velocity of the satellite in
% the ECF frame based on the COEs.  The input to this function is the COEs
% and time, and the output is the elements of the position vector and the
% velocity vector.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Declare global variables
global mu omega_earth gamma_g ar aw as
% Convert angles to radians
i = i*(pi/180); % convert i to radians
Omega = Omega*(pi/180); % convert RAAN to radians
w = w*(pi/180); % convert w to radians
nu = nu*(pi/180); % convert nu to radians

% Pre-define vectors and matrices for efficiency
m = length(a); % length of all vectors
p = zeros(m,1);
r = zeros(m,1);
r_p = zeros(3,1,m);
v_p = zeros(3,1,m);
R3_w = zeros(3,3,m);
R1_i = zeros(3,3,m);
R3_Omega = zeros(3,3,m);
```

```
R3_gamma = zeros(3,3,m);
r_ECF = zeros(3,1,m);
v_ECF = zeros(3,1,m);
r_ECI = zeros(3,1,m);
v_ECI = zeros(3,1,m);
x_ecf = zeros(m,1);
y_ecf = zeros(m,1);
z_ecf = zeros(m,1);
xdot = zeros(m,1);
ydot = zeros(m,1);
zdot = zeros(m,1);
x_eci = zeros(m,1);
y_eci = zeros(m,1);
z_eci = zeros(m,1);

% Use for loop to calculate r and v for the entire time interval
for j = 1:m
    % Calculate the semi-latus rectum
    p(j) = a(j)*(1-e(j)^2);
    % Calculate magnitude of radius vector
    r(j) = p(j)/(1+e(j)*cos(nu(j)));
    % write position vector in the perifocal frame
    r_p(:,:,j) = [r(j)*cos(nu(j));r(j)*sin(nu(j));0];
    % write the velocity vector in the perifocal frame
    v_p(:,:,j) = sqrt(mu/p(j))*[-sin(nu(j));e(j)+cos(nu(j));0];

    % Define rotation matrices to convert to ECI frame
    R3_w(:,:,j) = [cos(w(j)) -sin(w(j)) 0;
                   sin(w(j)) cos(w(j)) 0;
```

```
                0 0 1];
R1_i(:,:,j) = [1 0 0;
                0 cos(i(j)) -sin(i(j));
                0 sin(i(j)) cos(i(j))];
R3_Omega(:,:,j) = [cos(Omega(j)) -sin(Omega(j)) 0;
                   sin(Omega(j)) cos(Omega(j)) 0;
                   0 0 1];

% Define rotation matrix to convert to ECF frame
% Calculate Greenwich Local Sidereal time
gamma = gamma_g + omega_earth*(t(j)-t(1));
R3_gamma(:,:,j) = [cos(gamma) sin(gamma) 0;
                  -sin(gamma) cos(gamma) 0;
                   0 0 1];
% Perform coordinate transformation from perifocal to ECI frame
r_ECI(:,:,j) = R3_Omega(:,:,j)*R1_i(:,:,j)*R3_w(:,:,j)*r_p(:,:,j);
v_ECI(:,:,j) = R3_Omega(:,:,j)*R1_i(:,:,j)*R3_w(:,:,j)*v_p(:,:,j);
r_ECF(:,:,j) = R3_gamma(:,:,j)*r_ECI(:,:,j);
v_ECF(:,:,j) = R3_gamma(:,:,j)*v_ECI(:,:,j);
% Define variables
x_ecf(j) = r_ECF(1,1,j);
y_ecf(j) = r_ECF(2,1,j);
z_ecf(j) = r_ECF(3,1,j);
xdot(j) = v_ECF(1,1,j);
ydot(j) = v_ECF(2,1,j);
zdot(j) = v_ECF(3,1,j);
x_eci(j) = r_ECI(1,1,j);
y_eci(j) = r_ECI(2,1,j);
z_eci(j) = r_ECI(3,1,j);
```

```
end

% End of function file
```

Bibliography

1. Newberry, R. D., "Powered Spacefight for Responsive Space Systems," *High Frontier*, Vol. 1, No. 4, 2005, pp. 46–49.
2. Wertz, J. R., *Responsive Space Mission Analysis and Design*, Microcosm Inc., 2007.
3. Co, T., Zagaris, C., and Black, J., "Responsive Satellites Through Ground Track Manipulation Using Existing Technology," *AIAA Space 2011 Conference and Exposition*, No. AIAA-2011-7262, Long Beach, CA, September 2011.
4. "USSTRATCOM Space Control and Space Surveillance," `http://www.stratcom.mil/fUSSTRATCOM_Space_Control_and_Space_Surveillance`, Accessed 8 September 2011.
5. Kelecy, T. and Jah, M., "Detection and Orbit Determination of a Satellite Executing Low Thrust Maneuvers," *Acta Atronautica*, Vol. 66, 2009, pp. 798–809.
6. Wiesel, W. E., *Modern Orbit Determination*, Aphelion Press, 2010.
7. Folcik, Z. J., *Orbit Determination Using Modern Filters/Smoothers and Continuous Thrust Modeling*, Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge MA, 2008.
8. Payte, P. J., *Orbit Determination and Prediction for Uncorrelated Target Detection and Tracking*, Master's thesis, Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, 2011.
9. Sanchez, M. M. and Pollard, J., "Spacecraft Electric Propulsion - An Overview," *Journal of Propulsion and Power*, Vol. 14, No. 5, 1998, pp. 688–699.
10. Thorne, J. D., *Optimal Continuous-Thrust Orbit Transfers*, Ph.D. thesis, Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, 1996.
11. Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, Microcosm Press, 2007.
12. Conway, B. A., editor, *Spacecraft Trajectory Optimization*, Cambridge Univeristy Press, 2010.
13. Wiesel, W. E., *Modern Astrodynamics*, Aphelion Press, 2003.
14. Wiesel, W. E., *Spaceflight Dynamics (Third Edition)*, Aphelion Press, 2010.
15. Schaub, H. and Junkins, J. L., *Analytical Mechanics of Space Systems*, American Institute of Aeronautics and Astronautics, Inc., 2003.
16. Broucke, R. A. and Cefola, P. J., "On The Equinoctial Orbit Elements," *Celestial Mechanics*, Vol. 5, No. 3, 1972, pp. 303–310.

17. Kechichian, J. A., "Optimal Low-Thrust Rendezvous Using Equinoctial Orbit Elements," *Acta Astronautica*, Vol. 38, No. 1, 1996, pp. 1–14.

18. Chobotov, V. A., editor, *Orbital Mechanics (Third Edition)*, American Institute of Aeronautics and Astronautics, Inc., 2002.

19. Kirk, D. E., *Optimal Control Theory*, Dover Publications, 2004.

20. Bryson, Arthur E., J., *Dynamic Optimization*, Addison Wesley Longman, 1999.

21. Jacques, D., "Course Notes, MECH 622, Functional Optimization and Optimal Control," Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH.

22. Lawden, D. F., *Optimal Trajectories for Space Navigation*, Butterworth, 1963.

23. Marec, J.-P., *Optimal Space Trajectories*, Elsevier Scientific Publishing Company, 1979.

24. Alfano, S., *Low Thrust Orbit Transfer*, Master's thesis, Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, 1982.

25. Wiesel, W. E. and Alfano, S., "Optimal Many-Revolution Orbit Transfer," *Journal of Guidance, Control, and Dynamics*, Vol. 8, No. 1, 1985, pp. 155–157.

26. Hall, C. D. and Collazo-Perez, V., "Minimum-Time Orbital Phasing Maneuvers," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 6, 2003, pp. 934–941.

27. Ilgen, M. R., "Low Thrust OTV Guidance Using Lyapunov Optimal Feedback Control Techniques," *Advances in the Astronautical Sciences*, Vol. 85, Part 2, No. AAS 93-680, 1993, pp. 1527–1545.

28. Naasz, B. J., *Classical Element Feedback Control for Spacecraft Orbital Maneuvers*, Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, 2002.

29. Gurfil, P., "Nonlinear Feedback Control of Low-Thrust Orbital Transfer in a Central Gravitational Field," *Acta Atronautica*, Vol. 60, 2007, pp. 631–648.

30. Petropoulos, A. E., "Low-Thrust Orbit Transfers Using Candidate Lyapunov Functions with a Mechanism for Coasting," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, No. AIAA 2004-5089, Providence, RI, August 2004.

31. Yang, G., "Direct Optimization of Low-Thrust Many-Revolution Earth-Orbit Transfer," *Chinese Journal of Aeronautics*, Vol. 22, 2009, pp. 426–433.

32. Jean, I. and de Lafontaine, J., "Autonomous Guidance and Control of an Earth Observation Satellite Using Low Thrust," *Advances in the Astronautical Sciences*, Vol. 116, Part 3, No. AAS 03-617, 2003, pp. 1829–1844.

33. Guelman, M. and Kogan, A., "Electric Propulsion for Remote Sensing from Low Orbits," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 2, 1999, pp. 313–321.

34. Rao, A. V. et al., *User's Manual for GPOPS Version 4.X - A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Pseudospectral Methods*.

35. Escobal, P. R., *Methods of Orbit Determination*, John Wiley & Sons, Inc, 1965.

36. Larson, W. J. and Wertz, J. R., editors, *Space Mission Analysis and Deisgn*, Microcosm Press, 3rd ed., 2005.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE (DD–MM–YYYY)	2. REPORT TYPE	3. DATES COVERED *(From — To)*
22-03-2012	Master's Thesis	Jun 2011 – Mar 2012

4. TITLE AND SUBTITLE
Trajectory Control and Optimization for Responsive Spacecraft

5a. CONTRACT NUMBER
5b. GRANT NUMBER
5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Costantinos Zagaris, Capt, USAF

5d. PROJECT NUMBER
5e. TASK NUMBER
5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/ENY)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GA/ENY/12-M13

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Intentionally Left Blank

10. SPONSOR/MONITOR'S ACRONYM(S)
11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

The concept of responsive space has been gaining interest, and growing to include systems that can be re-tasked to complete multiple missions within their lifetime. The purpose of this study is to develop an algorithm that produces a maneuver trajectory that will cause a spacecraft to arrive at a particular location within its orbit earlier than expected. The time difference, Δt, is used as a metric to quantify the effects of the maneuver. Two separate algorithms are developed. The first algorithm is an optimal control method and is developed through Optimal Control Theory. The second algorithm is a feedback control method and is developed through Lyapunov Theory. It is shown that the two algorithms produce equivalent results for the maneuvers discussed. In-plane maneuver results are analyzed analytically, and an algebraic expression for Δt is derived. Examples are provided of how the analytic expression can be used for mission planning purposes. The feedback control algorithm is then further developed to demonstrate the simplicity of implementing additional capabilities. Finally, a set of simulations is analyzed to show that in order to maximize the amount of Δt achieved, a spacecraft must be allowed as much lead time as possible, and begin thrusting as early as possible.

15. SUBJECT TERMS
Control & Optimization; Optimal Control; Feedback Control; Lyapunov Theory

16. SECURITY CLASSIFICATION OF: UNCLASSIFIED			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	UU	136	Dr. Jonathan Black; Instructor, AFIT
U	U	U			19b. TELEPHONE NUMBER (Include Area Code) (937)255-3636, ext 4578 Email: jonathan.black@afit.edu

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18