The Pennsylvania State University

The Graduate School

MAXIMIZING PROPELLANT CONSUMPTION TO PREPARE AN IMPAIRED SATELLITE FOR RETRIEVAL

A Thesis in

Aerospace Engineering

by

Ashish Shenoy

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

December 2019

The thesis of Ashish Shenoy was reviewed and approved* by the following:

Robert G. Melton
Professor of Aerospace Engineering
Thesis Adviser

Puneet Singla
Associate Professor of Aerospace Engineering

Amy R. Pritchett
Professor of Aerospace Engineering
Head of the Department of Aerospace Engineering

*Signatures are on file in the Graduate School

Abstract

A man-made Earth satellite is a complex device consisting of a payload and a variety of subsystems that is placed into an orbit around Earth. Due to environmental factors and technological errors, the satellite's functions may become impaired which reduces or halts its ability to complete or continue its mission. An impaired satellite can either be left in orbit or could be retrieved by a retrieval mission if the satellite's stakeholder deems the satellite valuable enough to warrant repair. Any unused propellant remaining in the satellite must be used up to ensure a safe retrieval. It cannot simply be vented because it could create hazards for other satellites, such as adsorbing onto optical-sensor surfaces, but also because there is generally no means provided for venting unused propellants. The only other method to use up remaining propellant on a satellite is to turn on the thrusters for a specified burn period; however, doing so will change the orbit of the satellite which will require extra effort and resources to determine a suitable rendezvous point for a retrieval mission. This necessitates a solution that involves firing the satellite thrusters to completely use up all propellant without significantly changing the satellite's orbital characteristics.

This thesis used Particle Swarm Optimization (PSO) techniques to determine such a solution. PSO is an efficient meta-heuristic algorithm that uses swarm interactions and behavior to determine the best solution that fits the constraints of the problem it tries to solve. PSO was applied to the Gauss Variational equations to obtain the optimal time-variable thrust-angle functions that maximize the propellant use and minimize the change in the orbital elements of a satellite. 2-D and 3-D PSO formulations of the Gauss Variational Equations were tested on several sample orbits. Each orbit was tested in two scenarios, one where the burn time was user-fixed and one where the burn time was PSO-determined.

The PSO algorithm was able to determine optimal thrust-angle functions for all cases presented. The effectiveness of solutions gathered was demonstrated by the fact that each solution resulted in a final set of orbital elements equal to the initial set of orbital elements for each orbit to at least 10^{-12} accuracy while also consuming all of the propellant. The success of applying PSO methodology to minimize the change in orbital elements under specific conditions supports the notion that PSO can be applied to other types of specific or general orbital mechanics problems.

Table of Contents

List of Figures

List of Tables

List of Symbols

a	Semi-major axis
e	Eccentricity
in	Inclination
Ω	Right ascension of the ascending node
ω	Argument of periapsis
θ	True anomaly
E	Eccentric anomaly
p	Semi-latus rectum
r	Radial distance of satellite
h	Magnitude of angular momentum of satellite
μ	Gravitational parameter of the Earth-satellite system
G	Universal Gravitation Constant
m_1	Mass of the Earth
m_2	Mass of an artificial satellite
A_R	Magnitude of radial acceleration component of satellite
A_T	Magnitude of transverse acceleration component of satellite
A_N	Magnitude of orbit-normal acceleration component of satellite
A	Magnitude of total acceleration of satellite
δ	Thrust-angle in the orbital plane
φ	Thrust-angle normal to orbital plane
c	Effective exhaust velocity of satellite's propulsive system
n_0	Initial thrust-to-mass ratio of satellite
t_0	Thruster burn start time
t_f	Thruster burn end time
Δt	Thruster burn period
Δt_F	Maximum fixed thruster burn period used for normal burn
Δt_R	Burn period to consume remainder of propellant used for remainder burn
Δt_{total}	The total amount of time needed to consume all propellant
ε	Number of orbits to use a normal burn in
m_0	Initial mass of the satellite before burn
m_f	Final mass of the satellite after burn

f_F	Fuel consumed during a normal burn
f_R	Fuel consumed during remainder burn
δ_0	First coefficient of equation for δ
δ_1	Second coefficient of equation for δ
δ_2	Third coefficient of equation for δ
δ_3	Fourth coefficient of equation for δ
φ_0	First coefficient of equation for φ
φ_1	Second coefficient of equation for φ
φ_2	Third coefficient of equation for φ
φ_3	Fourth coefficient of equation for φ
t_N	Normalized time used in equations for φ
i	Index of particle
k	Index of element in a position, velocity, *GBest, PBest,* or *JBest* vector
n	The number of elements in a position, velocity, *GBest, PBest,* or *JBest* vector
P	The set of all particles used in PSO algorithm
N	The number of particles used
j	Iteration index
$P(i)$	Position vector for particle *i*
$V(i)$	Velocity vector for particle *i*
P_k	The *k*th element of a particle's position vector
V_k	The *k*th element of a particle's velocity vector
$P_k^{(j)}$	The *k*th element of a particle's position vector in the iteration *j*
$V_k^{(j)}$	The *k*th element of a particle's velocity vector in the iteration *j*
M	The total number of iterations
J	Objective function
$J^{(j)}(i)$	Value of objective function for particle *i* in iteration *j*
N_c	Current iteration number
c_I	Inertial coefficient
c_C	Cognitive coefficient
c_S	Social coefficient
h	Index of component of objective function
w_h	Weight for hth component of objective function
C_h	The hth component of the objective function

T_h	Threshold for hth to determine value of weight factor
W_h	Scalar value of the weight factor
H	Total number of components of J function
C_R	Component of objective function used in 3-D Time-Fixed formulation

Acknowledgements

I would like to express my thanks to my advisor, Professor Robert G. Melton, for his strong support, patient guidance and direction of this thesis. Dr. Melton is a superb teacher and his courses on Astrodynamics provided me with the necessary solid foundation for this research effort. Dr. Melton introduced me to the Particle Swarm Optimization technique and guided my research on this topic. I consider myself very fortunate to have been able to conduct my Master's thesis under such a highly respected and established expert in the Aerospace field.

I would like to thank my reviewers, Professor Puneet Singla and Professor Amy Pritchett, for taking the time to review this work and for providing me with valuable suggestions and advice. My thesis report reflects the results of their kind efforts.

I would to thank my parents for the love and support they have given me during my education and for all the guidance they have given me over the years.

Finally, I would like to thank the Graduate Admissions Office at Penn State University for allowing me this opportunity to study aerospace engineering and pursue a career in the aerospace industry.

CHAPTER 1. INTRODUCTION

1.1 Background

The launch of the first man-made satellite, Sputnik 1, galvanized the satellite-related aerospace industry and sparked an era of numerous technological and scientific developments in this field. Ever since then, satellites have become the backbone for a variety of industries servicing the commercial, government, and defense needs of many countries of the world. The satellites in use today serve a variety of purposes, such as fixed and mobile communications, TV distribution, weather forecasting, emergency and disaster communications, astronomy research, Earth observation, geo-imagery for intelligence, missile monitoring and launch detection, and navigation/global positioning, to name a few. Satellites are typically placed in a Low Earth Orbit (LEO), a Medium Earth Orbit, (MEO), or a Geostationary Orbit (GEO). A few satellites are placed in a Polar Earth Orbit (PEO) for coverage over the Polar regions. There are currently 5164 total satellites orbiting the Earth, 59 of which are in graveyard orbits [1].

Even though these modern satellites are culminations of rigorous and careful systems planning and engineering and are designed to be reliable with mission durations of at least 15 years, they can still be victims of malfunctions and failures that are a consequence of operating in the harsh space environment. As of March 31, 2019, there are a total of 2062 operational satellites [2]. This means that only about 40% of the existing satellites are functional, while the remaining roughly 60% are considered non-functional for various reasons. Satellites that are at the end of their operating life are typically moved to graveyard orbits or deorbited back to the Earth's atmosphere to be burned up upon reentry. The non-functional satellites likely experienced a failure that prevented them from fulfilling their intended mission. These non-functional satellites that remain in their orbits represent large investments of time and resources with no real return value to their stakeholders.

1.2 Satellite Malfunctions

The failure rate of satellites is a consequence of the materials and technology used as well as the surrounding spacecraft environment. At a high level, a typical modern satellite consists of its payload and its spacecraft bus components. The payload is usually a set of

electronic devices and sensors that enables the satellite to carry out its mission, while the spacecraft bus includes the structural, telemetry, power, thermal control and orbit/attitude control subsystems [3]. The components that comprise these systems are primarily electromechanical or electronic in nature. The failure events that can occur in electronics especially are partly due to the delicate nature of the materials used and physical processes that take place within the electronics. These types of failures are unavoidable and are usually minimized by shielding the electronics and incorporating redundancies and software recovery algorithms.

The surrounding spacecraft environment contributes to the majority of failure events in electronic components, regardless of how fault-tolerant and reliable the satellite is designed to be. The region of space containing satellites' orbits is exposed to radiation that comes from three main sources: the Van Allen Belts surrounding Earth, Galactic Cosmic Rays, and Solar Proton Events, otherwise known as Coronal Mass Ejections. The energetic photons, protons, electrons, ions, or heavier nuclei that radiation from these sources contains are particularly damaging to the sensitive electronics found on a satellite. Radiation affects both electronic devices and solar array cells. In addition, radiation can cause ionization of the atoms in electronic devices, which can generate its own current and when the radiation flux is high enough, this radiation current can swamp a device's current.

The space environment around Earth also contains micro-meteoroids (MM) and orbital debris (OD). Micro-meteoroids are naturally occurring while orbital debris is man-made. Both consist of particles that orbit the Earth at high velocities. Collisions in the MM/OD environment can cause physical and thermal damage to the satellite's surface as well as cause electromagnetic interference due to ionization formation. Orbital debris, in particular, is an ever-increasing problem because the total debris is growing at a rate of about 240 trackable pieces per year with a total count of over 20,000 objects orbiting the Earth as of 2017 [4]. This is because orbital debris is created from a variety of sources including non-operational spacecraft, spacecraft explosions of propellant tanks or batteries, collisions, and solid propellant and surface erosion particulates. Because about 60% of all orbiting satellites are non-operational, it is logical to conclude that the majority of OD added per year is derived from satellites that essentially should no longer remain in orbit. These non-operational satellites will continually contribute to the OD population as long as they

continue to orbit and this poses a hazard to all functioning satellites that still have a mission to perform.

There are several types of failures that can affect a satellite's ability to perform its mission. The physical hazards in the spacecraft environment, mentioned previously, are primary causes of the physical damage that a satellite can sustain which includes damage from collisions and from contamination. Contamination from OD particulates can disrupt instruments on the satellite and impair some, or all, of its functions. Collisions with MMs or ODs can damage major parts of the satellite or even destroy the entire satellite, resulting in breakups that create even more OD. Radiation can cause single event upsets (SEUs) in logic circuits and memory chips which can impact payload and certain subsystem functionality. Single event upsets occurs when a single energetic particle alters the operating characteristics of a device. This can lead to an "upset," in which the device functions in a manner inconsistent to its original design, and "latch-up," in which the device is transformed to an anomalous state that no longer responds to inputs. Both of these upset events may be permanent, which would render that device, and possibly the satellite as a whole, impaired. Energetic particles from radiation can also penetrate the satellite and build up charge inside the dielectric materials used inside. If this build-up of charge is not leaked or dissipated quickly enough, it can cause a dielectric discharge, which can trigger phantom commands or disruption or failure of the electronics used [5]. Additionally, a satellite that is not properly grounded will gain a charge build-up on the outside due to the electrons of the surrounding plasma and this can also cause a dielectric discharge. The surface charge also contributes to contamination because it causes the reattachment of outgassed particulates.

It is important to emphasize that, even if the payload is unharmed from these space hazards, the other subsystems of the satellite must also be unharmed enough to allow the satellite to perform essential functions such as communicating with ground stations or adjusting the attitude. If the payload is intact but the other subsystems are damaged, then the satellite could potentially be stranded in orbit. However, if the payload is damaged but the other subsystems remain operational, the satellite could be moved to another orbit where it can await retrieval or repair. This approach to salvage the satellite requires additional consideration and the impaired satellites that still have operational subsystems are the subject of this thesis. This thesis therefore addresses the problem of preparing an

impaired high-value satellite for repair or retrieval after it has experienced an event, either an MM/OD collision or an internal failure, that severely compromises its mission functionality, such as to warrant a repair or retrieval attempt.

1.3 Industry Efforts to Retrieve or Repair Malfunctioning Satellites

Most of the satellite industry is currently based on the premise that sending a satellite into orbit to perform a commercial or governmental service is an endeavor that has a zero chance of affording any satellite repair or maintenance in space. This means that the entire satellite system is designed with system redundancies to accommodate irrecoverable satellite malfunctions. The satellite itself will be designed to contain redundant parts, with more spares provided for critical subsystems than non-critical subsystems. The overall satellite system may also contain: (1) in-orbit spare satellites that remain on standby until the event of a catastrophic failure in the original satellite (2) redundant satellite coverage from other satellites in the constellation, and (3) spare satellites on the ground that could be launched at a later date in the event of an orbiting satellite's failure [6]. These redundancies entail a significant cost, with higher redundancies only possible for cheaper satellites or satellite constellations that reside in LEO. Satellites in GEO can cost up to $1.3 billion each, meaning it is prohibitively expensive to manufacture and keep redundant satellites on standby [7]. With the growing trend towards more technologically sophisticated and expensive satellites with complex communication payloads, it makes little economic sense to simply abandon a satellite that is malfunctioning.

There are currently no active technologies or missions to repair or retrieve satellites that are non-operational. Current efforts to do so are limited to the capabilities of ground control commands which might involve remotely resetting subsystems, switching on on-board redundant units, or commanding the spacecraft to perform necessary orbital maneuvers. An example of this occurred when the Canadian Telesat ANIK-E2 communications satellite experienced a gyroscopic guidance system failure and started spinning out of control. Rather than lose a $228 million satellite and estimated revenues of $3 billion, the Telesat satellite engineers analyzed sensor telemetry data and developed a solution that used the 22 thruster motors to reposition the spacecraft and restore service [8].

Strategies to salvage impaired satellites are the subject of ongoing research development and possible future deployment. NASA has been actively investigating in-orbit satellite repair and service missions for spacecraft besides the Hubble Space Telescope and International Space Station (ISS). NASA's Restore-L Mission, planned to launch in 2022, will demonstrate servicing technology for spacecraft in low-Earth orbit between 99 and 1,200 miles. These are the altitudes inhabited by the ISS, Hubble Space Telescope and various other Earth-observation and research satellites [9, 10, 11]. NASA is also developing the Robotic Refueling Mission (RRM) that intends to allow mission operators to remotely troubleshoot anomalies, investigate MM strikes, and perform teleoperated repairs. [12]. The Defense Advanced Research Projects Agency (DARPA) is working with Space Systems Loral (SSL) on the Robotic Servicing of Geosynchronous Satellites (RSGS) program, which will develop and demonstrate technologies that will enable cooperative inspection and servicing of GEO satellites, thus radically lowering the risk and cost of operating in GEO [9, 13]. These efforts show that there is an increasing demand for maintaining the operations of malfunctioning satellites. The research and development into this area must necessarily take into consideration additional factors that affect repair and retrieval missions of this nature.

1.4 Motivation: Propellant Hazard

The primary hazard to the repair or retrieval of satellite is the presence of unused propellant remaining in the propellant tanks of the satellite. This is because an on-orbit satellite servicing mission requires availability of the rescue vehicle, the availability of the launch rocket, and a suitable launch window. As a result, the impaired satellite needs to remain in orbit for potentially several months before it can be serviced. Unused propellant sitting in the tanks gives rise to the risk of a satellite breakup caused by explosion. Except for a few collisions in space amounting to fewer than 10 accidental and intentional events, the majority of the 200 satellite breakups observed in orbit to-date were due to explosions of the spacecraft and upper stages – due to leftover fuel, material fatigue or pressure increases in the batteries[14]. In the propulsion system, breakups occur due to internal pressure in the propellant and pressurant tanks, in the form of pressurant gas or residual propellants [14]. Due to the limited window of time available to act before the satellite power fails and commanding is no longer possible, it becomes imperative for the satellite owner/operator to immediately begin depletion burns to empty the propellant in the

thrusters while such commanding is still possible before the rescue satellite arrives. This is because any propellant left in the thruster' tanks represent a high explosion hazard, should any orbital debris collide with the tanks while the satellite is awaiting retrieval. A tank explosion would destroy the satellite completely, as well as pose a hazard to neighboring satellites, as opposed to a puncture in an empty tank, which is repairable. In addition, a tank explosion that occurs while the repair satellite is docked with the malfunctioning satellite would also destroy or severely damage the repair satellite, which would result in a huge financial loss for the satellite's owners.

There are two methods one might use to avoid catastrophic events caused by unused propellant. First, one can vent the fuel from the propellant tank and allow the satellite to maintain its current orbit. However, this is only possible for satellites designed to have this venting feature. Most satellites have no means to properly vent their propellant. Regardless, while this satisfies the need to void the propellant tanks as quickly as possible, it is not a viable option because the vented fuel becomes a collision hazard to the satellite itself when vented, or a collision hazard to neighboring satellites [15]. The only safe viable option is to consume all of the remaining propellant by firing the thrusters. However, such an operation imparts a change in velocity to the satellite, which will cause a change in the satellite's orbit. Any repair or retrieval satellite would need to dock with the impaired satellite and if the satellite's orbit has changed, then extra time and resources need to be spent to determine the satellite's new orbit and establish a rendezvous point.

Therefore, this thesis will tackle this problem by developing and demonstrating a method to determine the optimum thrust-angles a satellite should use so that its orbital elements are unchanged after a series of depletion burns. The methodology used here to determine such a solution will be to use particle swarm optimization, which is discussed in detail in Chapter 3. By employing the strategy presented in this thesis, a satellite stakeholder can effectively prepare the satellite for retrieval and repair without risking an explosion.

1.5 Literature Review

Previous research on this subject is limited to the work performed by Cichan et al [16]. The methodology therein combined control laws with the equations of motion of a

satellite in orbit and the time rate of change equations for the important orbital elements of an orbit. These control laws were used to minimize each orbital element's time derivative and thus minimize the change in orbital characteristics. Results from this study showed that minimization of the orbital elements was achieved with acceptable results with the sample orbit mostly unchanged in each of the four cases performed [16].

While the issue of expending unused propellant has not been appreciably pursued in research efforts, the problem of in-orbit satellite servicing, repair or retrieval has received much attention in the literature. The fundamental assumption in all these proposed methods is that it is much more economical to send robotic rescue satellites to the malfunctioning satellite than to send manned missions. Since future satellites are expected to be designed to be serviceable, with removable subassemblies and docking capabilities, robotic manipulators will need to be designed to avoid collision with its target with uninterrupted visual tracking. Zahroof et al presented one such design that uses remotely ground-controlled robotic arms and manipulators combined with high resolution cameras [17]. Similar robotic satellites with camera-aided stereo vision and multiple-DoF (degrees of freedom) robotic arms and docking mechanisms have been proposed by other researchers [18, 19, 20]. Additionally, Han and Hong [21] proposed a retrieval control strategy for a failed satellite that uses a servicing spacecraft with a tether attached. The servicing spacecraft would be a tethered space robot (TSR) or a tethered space tug (TST) that is used to capture and retrieve the target using an optimized tether swing angle [21].

The Particle Swarm Optimization (PSO) algorithm has been used in the literature to solve many orbital problems. One such example is the optimization of a maneuver orbit using multiple impulses for spatial operations involving non-coplanar and long-distance orbital maneuvers [22]. The proposed new hybrid algorithm combines Sequential Quadratic Programming (SQP) with PSO to reinforce the search accuracy and improve the convergence. Another application of PSO for solving spacecraft orbital problems is for the design of interplanetary trajectories with multiple synergistic gravitational assist maneuvers [23], where the PSO-optimized trajectories were compared with the real trajectories used by Voyager 1, Voyager 2, and Cassini to confirm their validity, and then applied to the problem of optimizing the multiple gravity assist (MGA) trajectory for a mission from Earth to Saturn. The PSO-based solution resulted in one half to one-third of the propellant cost compared to the direct routes for the launch and planned arrival dates.

Four cases of impulsive orbital transfers have been considered by Pontani and Conway for transfer trajectories between (a) two coplanar circular orbits, (b) two coplanar elliptic orbits with arbitrary inclination, (c) two non-coplanar circular orbits, and (d) two non-coplanar elliptic orbits. [24] The PSO algorithm was shown to result in effectively solving the orbital transfer problem with great numerical accuracy. Vtipil developed a methodology for determining the optimum orbital characteristics for a single satellite orbit around Earth where the total pass time and number of passes per day over 20 metropolitan areas were optimized to satisfy service. The PSO algorithm was used to maximize the satellite's visibility over the ground targets using two possible cost functions: one based on total pass time and another based on the average number of passes per day. The former cost function did not yield a unique solution, while the latter one did. The advantage of the PSO algorithm in either case was the much faster run time (within a few minutes) compared to a computer run time of a few hours using traditional methods [25]. Finally, Zhang et al used Particle Swarm Optimization to optimize the task allocations of a robotic service vehicle between client satellites. In this type of mission, the service vehicle would repair multiple malfunctioning satellites [26].

Transfer of a satellite from one Earth orbit to another Earth orbit has been studied before. Co-planar orbit transfers conducted by a servicing spacecraft to an impaired GEO satellite using the Hohmann Transfer approach has been analyzed using two models: a 2-body mechanics model and a High Precision Orbit Propagator (HPOP) model, from a perspective of designing a trajectory for observation of the target spacecraft prior to docking [27]. Orbit transfers using two possible methods of determining thrust direction and the timing of the thrusts have been analyzed by Petropoulos, where the goal was to effect specified changes in the orbital elements using the Gauss Variational Equations. The thrust, when on, is assumed constant with a constant specific impulse. Two examples of the transfers are provided, one in the semi-major axis and inclination, and one in the semi-minor axis and inclination. The control laws derived for the thrust profiles do not use the PSO technique and are based on a "time-to-go" concept, based on the times needed to affect the desired changes in the orbital elements, and a "proximity quotient" concept, based on an assessment of the proximity of the final orbit and time rate of change [28].

CHAPTER 2. PROBLEM DEFINITION

Before discussing how Particle Swarm Optimization is used to minimize the change in the orbital parameters of a satellite, it is necessary to describe the underlying physical model that the PSO algorithm will incorporate. This chapter will present the basic concepts needed as well as formally state the physical requirements of the problem that this thesis solves.

2.1 Orbital Elements and Derived Orbit Features

An orbit is defined as the trajectory of an object under the influence of a gravitational source. The gravitational source may be a sun, a planet, a moon, or even a black hole. This thesis will only consider the orbits of satellites around Earth, which fall in the category of orbits called planetary orbits.

A planetary orbit is characterized usually by a repeating regular motion of the orbiting object around a barycenter of the system. A barycenter is defined as the center of mass of a system containing two or more bodies. For Earth, the system under consideration will include the Earth and the man-made satellites that orbit it. Because the Earth's mass is extremely large compared to all the satellites, the barycenter is extremely close to the center of the Earth. For the purposes of this thesis, the satellites effectively orbit around Earth's center. The features of a generic orbit are described in the following paragraphs and are pictorially shown in Figure 2.1.

Figure 2.1: Example of Satellite Orbits around the Earth[1]

[1] Source: https://aerospace.csis.org/aerospace101/popular-orbits-101

Planetary orbits fall in the category of the Keplerian orbits, which means that they will all share the same set of defining features. The shapes of a Keplerian orbit can be circular, elliptical, parabolic, or hyperbolic depending on certain characteristics of the orbit. The planetary orbits pertinent to this thesis are elliptical or circular in nature. Every planetary orbit is formed around a point called the focus and the focus for every satellite's physical orbit around the Earth is said to be at the Earth's center.

An elliptical orbit contains two foci, with one called the primary focus and the other called the empty focus, while a circular orbit contains one focus which is usually referred to as the center. The gravitational source that an object orbits around will be located at the primary focus for an elliptical and at the center for a circular orbit.

Every planetary orbit has a periapsis, which is the point on the orbit that is closest in distance to the gravitational source. This naturally means that perfectly circular orbits have no periapsis because the orbit radius is constant. Planetary orbits also have an orbital plane, which intuitively, is the plane in three-dimensional (3-D) space that contains the foci and/or the center and the entire trajectory of the orbit. Most importantly, each orbit has six distinct parameters that uniquely identify it. These parameters are commonly called the Keplerian elements, or orbital elements. They are described below in relation to circular or elliptical orbits:

a) Semi-major axis length, denoted as a, is defined as one-half of the distance between the endpoints on the long axis of the ellipse. It is also referred to as the longest diameter of an elliptical orbit. In a circular orbit, the semi-major axis is equivalent to the radius of the circular orbit. This element has distance units in whatever measurement system is used.
b) Eccentricity, denoted as e, is a dimensionless measure of how much the orbit deviates from a perfect circular orbit. By definition, the eccentricity of a circular orbit is set to 0, while the eccentricity of an elliptical orbit is within the range $0 < e < 1$.
c) Inclination, denoted as in in this thesis and as i in Figure 2.2, is defined as the tilt of an object's orbit with respect to a plane of reference. For an Earth-centered orbit, the plane of reference is Earth's equatorial plane and the inclination is the angle between the orbital plane of the satellite and the Earth's equator. The inclination is

measured in degrees or radians and has a value within the range $0 \leq in \leq 180$ degrees or $0 \leq in \leq \pi$ radians.

d) Right Ascension of the Ascending Node (RAAN), denoted as Ω, is the angle between the ascending node's "node vector" and the line connecting the Earth's center and the First Point of Aries. The First Point of Aries (FPA) is a reference used in celestial coordinate systems and is also the location of the vernal equinox. The ascending node is the point at which the satellite crosses from the Earth's southern hemisphere to the Earth's northern hemisphere. This is possible only if the inclination is greater than 0. The RAAN is measured counterclockwise from the FPA as seen from North and has units of degrees or radians. Values of the RAAN are within the range $0 \leq \Omega < 360$ degrees or $0 \leq \Omega < 2\pi$ radians.

e) Argument of Periapsis (AOP), denoted as ω, is the angle between the ascending node's "node vector" and the eccentricity vector. The eccentricity vector points from the primary focus to the periapsis. Thus, the argument of periapsis lies in the orbital plane. The argument of periapsis has units of degrees or radians and is measured in the direction of the satellite motion starting from the node vector. Its value lies within the range of $0 < \omega < 180$ degrees ($0 < \omega < \pi$ radians) if the eccentricity vector points above the equatorial plane and $-180 < \omega < 0$ degrees (-$\pi < \omega < 0$ radians) if the eccentricity points below the equatorial plane.

f) True Anomaly, denoted as θ, is the angle between the eccentricity vector and the position vector of the object, with both vectors starting at the focus. For a circular orbit, the true anomaly is measured from a reference point on the orbit where θ is set to 0. It is measured in the direction of the object's motion and is in the range of $0 \leq \theta < 360$ degrees or $0 \leq \theta < 2\pi$ radians. The value of theta θ at some reference time serves as the sixth orbital element.

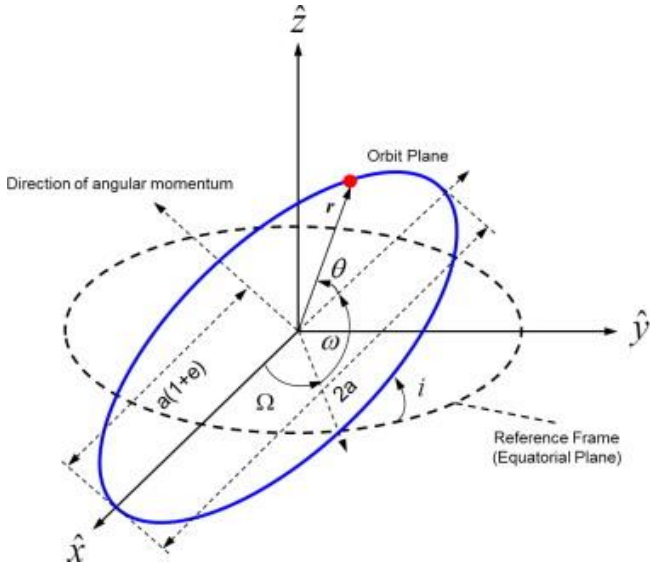

Figure 2.2: Orbital Elements of a Keplerian Orbit around Earth[2]

Although the above orbital elements are required for defining an orbit, there are other useful quantities related to orbits that can be derived from the main orbital elements. These additional parameters are used in the physical model of this thesis and are described below:

a) Eccentric Anomaly, denoted as E, like the true anomaly, is another measure of an object's position in its orbit. Like the true anomaly, it is measured in the direction of motion from the periapsis. The difference is that eccentric anomaly uses the center of the ellipse rather than the focus. The eccentric anomaly is related to the true anomaly as follows:

$$\tan\frac{\theta}{2} = \sqrt{\frac{1+e}{1-e}}\tan\frac{E}{2} \tag{1}$$

For a circular orbit, the eccentricity e = 0, which means that the eccentric anomaly is equivalent to the true anomaly. Eccentric anomaly has the same units and range as true anomaly.

[2] Source: https://www.researchgate.net/figure/Definition-of-Keplerian-elements-The-orbital-elements-to-describe-the-trajectory-of_fig4_272198427

b) Semi-latus Rectum, denoted as p, is defined as half of the length of the chord that passes through the focus and that has end points on the ellipse. It is perpendicular to the long axis of an ellipse. The equation is given below:

$$p = a(1 - e^2) \tag{2}$$

The semi-latus rectum for a circular orbit is thus equal to the radius.

c) Radial distance, denoted as r, is intuitively the straight-line distance between the primary focus and the object that orbits around it. It uses distance units and is given by:

$$r = \frac{p}{1 + e\cos\theta} \tag{3}$$

d) Angular momentum magnitude, denoted as h, is given by the equation:

$$h = \sqrt{\mu p} \tag{4}$$

where μ is the gravitational parameter of the system, defined as $\mu = G(m_1 + m_2)$ in a two-body system. In this case, the constants m_1 and m_2 represent the masses of the Earth and artificial satellite, respectively, while the constant G is the Universal Gravitation Constant. Clearly, $m_1 \gg m_2$, which means $\mu \approx Gm_1$.

This concludes the summary of the basic elements of a satellite's orbit around Earth. The next section will present the set of equations that ties all the orbital elements together.

2.2 Gauss Variational Equations and the Satellite Thrust Model

The main set of equations that this thesis relies on are known as the Gauss Variational Equations (GVEs), which relates the time rate of change of each of the six major orbital elements $(a, e, in, \Omega, \omega, \text{and } \theta)$ in terms of the other orbital elements and and external non-gravitational forces. The GVEs are presented below [29]:

$$
\begin{aligned}
\frac{da}{dt} &= \frac{2a^2}{h}\left(A_R e \sin\theta + A_T \frac{p}{r}\right) \\
\frac{de}{dt} &= \frac{1}{h}\{A_R p \sin\theta + A_T[(p+r)\cos\theta + re]\} \\
\frac{d(in)}{dt} &= A_N \frac{r\cos(\theta+\omega)}{h} \\
\frac{d\Omega}{dt} &= A_N \frac{r\sin(\theta+\omega)}{h\sin(in)} \\
\frac{d\omega}{dt} &= \frac{1}{he}[-A_R p\cos\theta + A_T(p+r)\sin\theta] - A_N \frac{r\sin(\theta+\omega)\cos(in)}{h\sin(in)} \\
\frac{d\theta}{dt} &= \frac{h}{r^2} + \frac{1}{eh}[A_R p\cos\theta - A_T(p+r)\sin\theta]
\end{aligned}
\tag{5}
$$

These equations include the additional quantities A_R, A_T, and A_N, which are the acceleration components in the radial, transverse and normal directions, respectively. These represent the components of acceleration of an object due to a perturbing force and these acceleration components will affect the rate of change of the orbital elements. In this case, the perturbed acceleration of the satellite is due to the force from the thrusters. For the purposes of this thesis, any other perturbing force acting on the satellite is effectively negligible compared to the thruster force. The definitions of each of these accelerations are listed below:

1. A_R is the magnitude of the radial acceleration vector that points in the radial direction parallel to the radial unit vector $\hat{e}_R$. The radial unit vector points outward from the focus to the satellite. The equation of the radial acceleration is defined as:

$$A_R = A\cos\varphi\sin\delta \tag{6}$$

2. A_T is the magnitude of the transverse acceleration vector that points in the transverse direction parallel to the transverse unit vector $\hat{e}_T$. The $\hat{e}_T$ vector always points perpendicular to the $\hat{e}_R$ in the direction of increasing θ. The equation of the transverse acceleration is defined as:

$$A_T = A\cos\varphi\cos\delta \tag{7}$$

3. A_N is the magnitude of the normal acceleration vector that points normal to the orbital plane parallel to the normal unit vector $\hat{e}_N$. The equation of the normal acceleration is defined as:

$$A_N = A \sin \varphi \tag{8}$$

The quantity A is the magnitude of the total thrust-acceleration vector of the satellite, while the angles φ and δ specify the total thrust-acceleration vector's orientation in 3-D space with respect to the orbital plane as shown in Figure 2.3. The angle δ is measured from local horizontal plane in the direction of transverse motion and has a range of $0 \leq \delta < 2\pi$ radians. The angle φ is measured such that $\varphi = 0$ when the total thrust-acceleration is entirely in the orbital plane. This angle is in the range of $-\frac{\pi}{2} \leq \varphi \leq \frac{\pi}{2}$ where positive values lie above the orbital plane and negatives values lie below the orbital plane. Thus, δ specifies components of acceleration in the orbital plane, while φ specifies the components normal to the orbital plane.

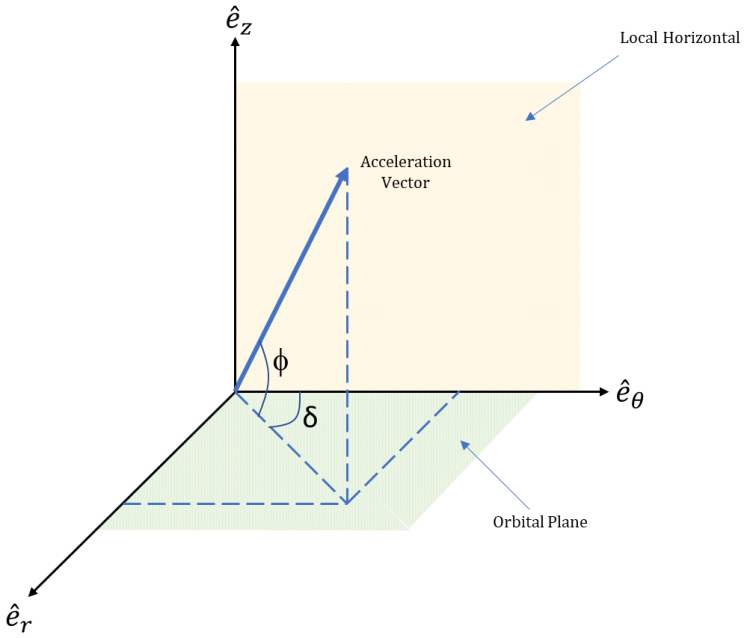

Figure 2.3: δ and φ with Respect to Orbital Plane and Local Horizontal

The total acceleration magnitude itself is given by:

$$A = \frac{cn_0}{c - n_0 t} \tag{9}$$

where c is the effective exhaust velocity of the satellite's propulsive system, n_0 is the thrust-to-mass ratio of the satellite at an initial time t_0, and t is the time within the burn period Δt where $t_0 \leq t \leq t_f$ and $\Delta t = t_f - t_0$ for some final time t_f.

The variational equations by themselves can describe a general 3-D orbit behavior over time; however, these equations can be reduced to describe 2-D behavior as well. The key to doing this involves setting the orbit normal angle φ to 0, meaning that the satellite's thrusters will only fire along the plane of the orbit and thus the satellite's acceleration will be only in orbital plane. This sets that the normal acceleration component $a_N = 0$ which then reduces the GVEs to:

$$\begin{aligned}
\frac{da}{dt} &= \frac{2a^2}{h}\left(A_R e \sin\theta + A_T \frac{p}{r}\right) \\
\frac{de}{dt} &= \frac{1}{h}\{A_R p \sin\theta + A_T[(p + r)\cos\theta + re]\} \\
\frac{d(in)}{dt} &= 0 \\
\frac{d\Omega}{dt} &= 0 \\
\frac{d\omega}{dt} &= \frac{1}{he}[-A_R p \cos\theta + A_T(p + r)\sin\theta] \\
\frac{d\theta}{dt} &= \frac{h}{r^2} + \frac{1}{eh}[A_R p \cos\theta - A_T(p + r)\sin\theta]
\end{aligned} \tag{10}$$

Thus, the orbital characteristics will change only within the confines of the orbital plane, which is a 2-D plane. The inclination and RAAN are elements that describe an orbit's orientation with respect to the equatorial plane and a change in them will affect the orbit's 3-D orientation with respect to the equatorial plane. The orbit normal component of the AOP can also affect the 3-D orientation of an orbit. This formulation of the GVEs is useful because it reduces the search space of the PSO algorithm and can result in better and faster converging results. However, both 2-D and 3-D versions are viable approaches to this orbital problem and both are employed in this thesis. They require different approaches to

the implementation of the particles in the algorithm and the formulation of the thrust angle of the satellites, as described below.

In order to use up the propellant without changing orbit, the final orbital elements of the orbit after the burn period must be equal to the orbital elements at the beginning of the burn period. This means that the satellite must continually angle its axial thrusters throughout the burn period in a specific way to make this possible. Note that the satellite is allowed to deviate as necessary from its original orbit during the burn period and this is to be expected. To achieve this, the thrust-angles as a function of time will be determined by the PSO algorithm used here. Although the angles δ and φ were defined in relation to the acceleration components, they serve as the thrust-angles as well because the thrust force acts in the same direction as the acceleration vector.

The burn period of the satellite is another variable that is important to developing the solution approach. Most satellite thrusters have only two settings: (1) completely off or (2) turned on at maximum thrust. Due to physical limitations of a satellite's thrusters or propulsion system, a satellite with thrusters turned on can keep the thrusters turned on for only a limited time before the thrusters must be turned off or are automatically turned off to prevent overheating and damage to the thruster itself. This means that the maximum burn period a satellite can have during a maneuver here will be some finite fixed period Δt_F. Since the goal is to maximize propellant consumption during each burn, turning thrusters on for the full Δt_F will consume as much propellant as possible for that burn. Unfortunately, for the impaired satellites this thesis discusses, Δt_F is likely not enough to fully consume the remaining propellant. Additionally, Δt_F is also much shorter than the period of a satellite's orbit. This means that a satellite will need to do multiple fixed burns in multiple orbits to consume all the propellant. The most expedient approach would be to use up the majority of propellant using several burns lasting Δt_F and then perform a final burn to use up the remaining propellant that would require a burn period shorter than Δt_F. Let Δt_{total} be the total time needed to consume all remaining propellant in the satellite. Then the relation between Δt_{total} and Δt_F is:

$$\Delta t_{total} = \varepsilon \Delta t_F + \Delta t_R \tag{11}$$

where ε is the number of orbits where a burn lasting Δt_F is used and Δt_R is the burn period for any leftover propellant. Therefore, the PSO algorithm needs to determine the thrust-angle equations for a burn lasting Δt_F, which this paper will refer to as "normal burn", and the thrust angle equations for a burn lasting Δt_R, which this paper will refer to as "remainder burn". The total time needed to burn all the propellant is given by:

$$\Delta t_{total} = \frac{c}{n_0}\left(1 - \frac{m_f}{m_0}\right) \tag{12}$$

where m_0 is the initial mass of the satellite before performing the burn and m_f is the final mass of the satellite after all propellant is depleted. The fuel consumed during the normal burn is:

$$f_F = \left(\frac{n_0}{c}\right)\Delta t_F \tag{13}$$

The fuel consumed during the remainder burn can be calculated by:

$$f_R = \left(\frac{n_0}{c}\right)(\Delta t_{total} \bmod \Delta t_F) \tag{14}$$

With this information in mind, the thrust-angle equations will be described as follows. Each angle δ and φ is modeled as a 3rd order polynomial. In the 2-D simulations, the orbital-plane angle has the form:

$$\delta = \delta_0 + \delta_1 t + \delta_2 t^2 + \delta_3 t^3 \tag{15}$$

where the coefficients $\{\delta_0, \delta_1, \delta_2, \delta_3\}$ are unknowns that the PSO algorithm will determine. In the 3-D cases, the orbital-plane angle has the same form as Equation (15), but the orbit-normal angle has the form:

$$\varphi = \frac{\pi}{8}[\varphi_0 T_0(t_N) + \varphi_1 T_1(t_N) + \varphi_2 T_2(t_N) + \varphi_3 T_3(t_N)] \tag{17}$$

where the coefficients $\{\varphi_0, \varphi_1, \varphi_2, \varphi_3\}$ are unknowns that the PSO algorithm will also determine, T_0, T_1, T_2, T_3 are Chebyshev polynomials of the first kind of orders 0, 1, 2, and 3,

and t_N is the normalized time of the burn. This time is normalized to help constrain the value of φ to within $\left[-\frac{\pi}{2}\ ,\ \frac{\pi}{2}\right]$. The normalized time is defined as:

$$t_N = \frac{2}{\Delta t} t - 1 \tag{18}$$

where $\Delta t = \Delta t_F$ for the normal burn and $\Delta t = \Delta t_R$ for the remainder burn.

Thus, the goal of the PSO algorithm applied in this thesis is to solve for the coefficients defined above for both normal burn and remainder burn using either 2-D or 3-D formulations. Achieving this goal will demonstrate that PSO is very suitable for solving this type of orbital problem. At this point, it is important to note that the GVE can also be used to determine the satellites position and velocity at any point in the orbit. Of particular interest to the satellite owners/operators or ground control staff is the position of the satellite in the Earth-Centered Inertial (ECI) frame as it performs its burn. This is solved using the equations below:

$$\begin{gathered} \bar{\boldsymbol{r}}^{\boldsymbol{P}} = (r\sin\theta)\widehat{\boldsymbol{P}} + (r\cos\theta)\widehat{\boldsymbol{Q}} + 0\widehat{\boldsymbol{W}} \\ \bar{\boldsymbol{r}}^{\boldsymbol{E}} = C_{\boldsymbol{PE}}\bar{\boldsymbol{r}}^{\boldsymbol{P}} \\ C_{\boldsymbol{PE}} = [C_3(\omega)C_1(in)C_3(\Omega)]^T \end{gathered} \tag{19}$$

where $\bar{\boldsymbol{r}}^{\boldsymbol{P}}$ is the satellite's position vector in the Perifocal frame, $\bar{\boldsymbol{r}}^{\boldsymbol{E}}$ is the satellite's position vector in the ECI frame, and $C_{\boldsymbol{PE}}$ is the direction cosine matrix that maps a vector in the Perifocal frame to the ECI frame. The direction cosine matrices C_1 and C_3 area defined as:

$$C_3(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{20}$$

$$C_1(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{21}$$

where α is the angle that the coordinate system rotates by. Using these equations, one can plot the position of the satellite across all the iterations of the algorithm, which is useful for visualizing the motion of the satellite as it gets perturbed by the thrusters during the burn period.

CHAPTER 3. PARTICLE SWARM OPTIMIZATION

3.1 Background

Particle Swarm Optimization (PSO) is a metaheuristic optimization algorithm that can be used to determine a sufficiently optimal solution to a multivariate problem. It is inspired by the group behavior of swarms of animals or creatures found in nature, such as birds, fish, or bees. Each member of the swarm interacts with other members of the swarm to collectively achieve some goal which, in nature, usually involves searching for food. Even though the interactions within the swarm between members are mostly unpredictable, the information shared between members shapes the overall behavior of the swarm as well as the individual behavior of each member. This is indicated by how all members of the swarm eventually reach the same goal with no stragglers. In the computational world, this swarm behavior is modelled in the PSO algorithm and reflected in the results achieved when using it.

PSO is a suitable optimization algorithm that can be applied to a variety of problems in many different disciplines. This is partly because, as previously mentioned, PSO is a metaheuristic algorithm. A metaheuristic algorithm is a type of high-level heuristic algorithm that is generic enough to be applicable to many distinct problems that would otherwise require more specific heuristic algorithms. A metaheuristic algorithm also attempts to locate the global optimum in a global search space that contains many local optima, while a heuristic algorithm attempts to locate a local optimum in a more specific search space. PSO can achieve this by employing an element of randomness, but this comes at the cost of possibly finding sub-optimal solutions [30]. This problem is easily overcome either by modifying the basic PSO algorithm to model a different type of swarm behavior that provides better results, by using a hybrid approach where the PSO results are used as an initial “best guess” for a different optimizer [30, 31], or by simply re-running the PSO algorithm to hopefully compute a better solution. This weakness is far outweighed by the strengths of PSO, which include its simple design and implementation, its ease of applicability without changing the core features, its robustness, and its typically fast convergence speed [32, 33]. These strengths have made PSO very popular ever since its creation in 1995 by Kennedy and Eberhart [34], and the algorithm has seen use in problems

involving communication satellite design [35], power and voltage control [36], neural networks [37], industrial orienteering [32], and spacecraft orientation [30].

In addition, researchers have taken inspiration from PSO theory to create modified versions of PSO that are based on different natural phenomena. For example, Pluhacek et al [38] created a PSO based metaheuristic based on the gathering behavior of humans where each particle's position remains stationary for a while when the particle finds promising solution behavior, thus allowing other particles to "notice" and "follow" and create a larger group of possible solutions in a small space. There also exist algorithms based on the genetic operations such as selection, crossover, and mutation, and based on immune system features such as antibody behaviors and immune memory. Other PSO-inspired algorithms are modeled on differential evolution, ant colony behavior, and bacterial foraging [23]. These modifications are aimed at eliminating PSO's shortcomings, such as slow convergence and entrapment in local optima, when it is used in very complex problems. However, for the purposes of this thesis, modification of the PSO algorithm was not necessary because the basic form of the algorithm was sufficient in determining the best solutions within a reasonable time frame. The basic PSO algorithm is described in the following paragraphs.

3.2 General PSO Algorithm Details

Each feature of the swarms found in nature has its computational counterpart modelled in the PSO algorithm. Each creature in the swarm is modelled as a "particle" in the algorithm. A particle is an entity that has a "position" vector and a "velocity" vector associated with it. The position vector of any particle i is defined as:

$$\boldsymbol{P}(i) \triangleq [P_1(i) \ \dots \ P_k(i) \ \dots \ P_n(i)] \quad \text{where} \quad 1 \leq k \leq n \tag{22}$$

where $\boldsymbol{P}(i)$ is the position vector of particle i containing n elements and $P_k(i)$ is the kth element of this position vector. The position of a particle does not refer to any physical position. Instead, it refers to the set of values that correspond to one possible solution of the multivariate problem under investigation. Each particle will thus contain a possible solution in its position vector and each particle will share information related to its position vector with the other particles. The position vectors for the entire swarm are therefore the set of all position vectors:

$$\boldsymbol{P} = [P(1) \ \ldots \ P(N)]^T \quad \text{where} \quad 1 \leq i \leq N \tag{23}$$

where *N* is the number of particles in the swarm. The natural world analogue of the position vector would be the knowledge held by an individual creature of the swarm about the possible location of food or prey, with each member knowing one possible location.

The velocity vector of any particle *i* is similarly defined as:

$$\boldsymbol{V}(i) = [V_1(i) \ \ldots \ V_k(i) \ \ldots \ V_n(i)] \quad \text{where} \quad 1 \leq k \leq n \tag{24}$$

where $\boldsymbol{V}(i)$ is the velocity vector of particle i containing n elements and $V_k(i)$ is the kth element of this velocity vector. The velocity vector of a particle refers to the rate of change of its position vector per iteration of the algorithm. The particle uses each V_k of its velocity vector to update each corresponding P_k of its position vector by adding the V_k to the current P_k:

$$P_k^{(j+1)} = P_k^{(j)} + V_k^{(j)} \quad \text{where } 1 \leq j \leq M \tag{25}$$

where $P_k^{(j)}$ and $V_k^{(j)}$ is the value at the *j*th iteration of the *k*th element of the position and velocity vectors, respectively, of the *i*th particle of the swarm, and M is the total number of iterations. Then $P_k^{(j+1)}$ is the particle's updated position element of its position vector, which will be used in the (*j+1*)th iteration. This represents the particle updating its own information on possible values of the final optimal solution. Note that the particle's "knowledge" of the optimal solution is updated in every iteration, which models the dynamic nature of real-life creature swarms.

The PSO algorithm uses the position vectors of each particle to determine the optimal solution by using an objective function J. During each iteration, the algorithm will compute a J value using each particle's position vector. This means that the set of J values computed during a single iteration can be represented as:

$$J^{(j)}(i) = J(P^{(j)}(i)) \qquad (i = 1, \ldots, N) \tag{26}$$

where $J^{(j)}(i)$ is the J computed at the *j*th iteration for the *i*th particle and $J(P^{(j)}(i))$ is the objective function that uses as its parameter the position vector in the *j*th iteration for the

*i*th particle. The PSO algorithm selects the minimum J value calculated during one iteration and uses it to determine both the global best J value *GG* and its corresponding global best position vector *gBest*:

$$GG(j) = \min\left(J^{(j)}(i)\right) \ (i = 1, \dots, N)$$
$$GBest = GBest(j) = GBest(GG(j)) \tag{27}$$

where j is the current iteration. The $GBest$ vector and GG value represent information shared between all particles needed to adjust their knowledge of the optimal solution, i.e., to adjust their position vectors. The global best J and global best positions are updated after each calculation of J values. In addition, each particle keeps track of the best J value that it has achieved up until and including the current iteration. This is defined as:

$$JBest(i) = \min(J^{(j)}(i)) \qquad (j = 1, \dots, N_C) \tag{28}$$

where N_C is the value of j for the current iteration. The value *JBest* is used to determine the vector *PBest,* which is a particle's position vector that led to determining *JBest*. The vectors *GBest* and *PBest* for the current iteration j are used to update the velocity vector for each particle using the following formula:

$$V_k^{(j+1)}(i) = c_I V_k^{(j)}(i) + c_C\left[PBest_k^{(j)}(i) - P_k^{(j)}(i)\right] + c_S\left[GBest_k^{(j)} - P_k^{(j)}(i)\right] \tag{29}$$

where c_I, c_C, and c_S are known as the inertial, cognitive, and social coefficients. The inertial coefficient contributes to keeping the velocity unchanged, the cognitive coefficient contributes to adjusting the particle's position towards its *PBest*, and the social coefficient helps to adjust the particle's position towards the global *GBest*.

Thus, the velocity used to determine the position for the next iteration will be influenced by the particle's attempt to reach the best possible solution, which can either be its current known solution, a solution that it has reached in the past, or a solution that the swarm collectively has reached. These coefficients are defined as follows:

$$c_I = \frac{1}{2}\left(1 + r_1(0,1)\right) \qquad c_C = 1.49445 r_2(0,1) \qquad c_S = 1.49445 r_3(0,1) \tag{30}$$

where $r_1(0,1), r_2(0,1)$, and $r_3(0,1)$ are three independent random number generators that give a random number between 0 and 1 with uniform distribution. The new position will then be updated using Equation (25) and Equation (29). The value of each $P_k(i)$ and $V_k(i)$ will lie in a range bounded by an upper and lower bound:

$$\begin{aligned} B_{L,P_k} \leq P_k \leq B_{U,P_k} \\ B_{L,V_k} \leq V_k \leq B_{U,V_k} \end{aligned} \tag{31}$$

Note that values of the upper and lower bounds are not dependent on any individual particle and depend only on the nature of the element in the kth position of the position and velocity vectors. These bounds must be enforced when velocities and positions are calculated and updated. If the velocity bounds are violated, the velocity is to:

$$\begin{aligned} V_k^{(j+1)} = B_{L,V_k} \quad if \quad V_k^{(j+1)} < B_{L,V_k} \\ V_k^{(j+1)} = B_{U,V_k} \quad if \quad V_k^{(j+1)} > B_{U,V_k} \end{aligned} \tag{32}$$

If the position bounds are violated, the position and velocity are instead set to:

$$\begin{aligned} P_k^{(j+1)} = \begin{Bmatrix} B_{L,P_k} & if & P_k^{(j+1)} < B_{L,P_k} \\ B_{U,P_k} & if & P_k^{(j+1)} > B_{U,P_k} \end{Bmatrix} \\ V_k^{(j+1)} = 0 \quad if \quad V_k^{(j+1)} < B_{L,V_k} \quad or \quad V_k^{(j+1)} > B_{U,V_k} \end{aligned} \tag{33}$$

Putting these features together gives the algorithm shown in Figure 3.1:

PSO_Basic_Algorithm

for $j = 1 : M$

$\boldsymbol{J} = J(\boldsymbol{P})$

if $\boldsymbol{J}(i) < \boldsymbol{JBest}(i)$

$\boldsymbol{PBest}(i) = \boldsymbol{P}(i)$

$\boldsymbol{JBest}(i) = \boldsymbol{J}(i)$

if $\boldsymbol{J}(i) < GG$

$GG = \boldsymbol{J}(i)$

$GBest = \boldsymbol{P}(i)$

$c_I = \frac{1+r_1(0,1)}{2}$; $c_C = 1.49445 r_2(0,1)$; $c_S = 1.49445 r_2(0,1)$;

for $i = 1 : N$

$\boldsymbol{V}(i) = c_I \boldsymbol{V}(i) + c_C[\boldsymbol{PBest}(i) - \boldsymbol{P}(i)] + c_S[\boldsymbol{GBest} - \boldsymbol{P}(i)]$

for $k = 1 : n$

if $V_k(i) < B_{L,V_k}$

$V_k(i) = B_{L,V_k}$

else if $V_k(i) > B_{U,V_k}$

$V_k(i) = B_{U,V_k}$

end

for $i = 1 : N$

$\boldsymbol{P}(i) = \boldsymbol{P}(i) + \boldsymbol{V}(i)$

for $k = 1 : n$

if $P_k(i) < B_{L,P_k}$

$P_k(i) = B_{L,P_k}$

$V_k(i) = 0$

else if $P_k(i) > B_{U,P_k}$

$P_k(i) = B_{U,P_k}$

$V_k(i) = 0$

end

end

end

return $GBest$

end

Figure 3.1: Visual Outline of the Generic PSO Algorithm

3.3 PSO Adjustments Specific to Orbital Elements Minimization

As stated previously, the basic PSO algorithm will be applied here. The purpose of this is to demonstrate that PSO concepts can be used to solve a specific orbital problem that primarily uses the GVEs. Thus, no major modifications to the core algorithm were used. This means that the PSO simulations here provide a benchmark demonstrating what is possible under basic conditions, which can then be used to develop, and compare with, more refined techniques in future research. It also means that the actions undertaken in this thesis are exploratory in nature, with emphasis placed on finding an acceptable PSO solution, rather than rigorously testing certain PSO approaches.

To provide a more robust coverage of PSO capability with the GVEs, two types of simulations were run. These simulations involve the selection of the maximum burn period mentioned earlier, which is the maximum time a satellite can keep its thrusters on at maximum thrust before needing to shut off the thrusters. Each satellite will have a maximum burn period and each satellite's burn period will be different, owing to differences in thruster design, construction, or materials. Although firing thrusters for maximum time will always consume fuel in the most efficient manner, it may not result in proper minimization of orbital elements. Therefore, the option must be allowed for PSO to choose the proper burn period within some range that would strike a balance between achieving minimization while also using as much fuel as possible per orbit. Choosing a burn period within some range will have the added effect of attaining solutions for different burn periods, which would represent the different possible maximum burn periods of satellites currently in use. Thus, simulations were run where the burn period was user-fixed, referred to here as "Time-Fixed" and where the PSO algorithm determined the burn period, referred to here as "Time-Variable". Since the burn period for the remainder burn is known after determining or knowing the burn period for the normal burn, the thrust-angle equation for remainder burns can be calculated using the Time-Fixed version of the algorithm. Simulations where the normal burn thrust equations were calculated by a Time-Fixed PSO algorithm will be referred to in this paper as "Timed-Fixed cases", while simulations that calculate normal burn thrust equations using Time-Variable PSO will be referred to as "Time-Variable cases".

To help with convergence speed and convergence results, an addition to the J function was used. As previously mentioned, the J function is used to determine how optimal a particle's solution is compared to other particles. The criteria in this thesis for the "goodness" of a solution is how well it minimizes the change in orbital elements of the satellite's orbit before the burn maneuver and after the burn maneuver. Additionally, for certain cases, the J function will include a minimization of the fuel remaining after a burn. Paired with each J function component C is a weight value. These weights serve to increase the magnitude of the J function component, which will in turn increase the value of the J function. While this might seem counterintuitive, the effect is that it forces the particles of the PSO algorithm to search for better solutions that avoid increasing the J function value. These weights therefore are similar in function to the weights used for the J function in Cichan et al's work [16].The general form of the J function used in all the simulations run is in the form:

$$J = \sum_{h=1}^{H} w_h C_h \tag{34}$$

where w_h is the weight. The weights are defined as follows:

$$w_h = \begin{cases} W_h & if\ C_h > T_h \\ 1 & if\ C_h < T_h \end{cases} \tag{35}$$

where W is a positive scalar value and T is the threshold value the algorithm uses to decide what weight value to use.

An additional technique to attain better solutions is the process of modifying the position bounds. A user can adjust the position bounds based on the output J values and the final particle element values. If a certain particle achieves a promising result after a completed simulation, the bounds can be adjusted based on a neighborhood around each particle element and the simulation can be re-run using the adjusted bounds. This effectively reduces the search space the algorithm works with and increases the chance of finding the optimal solution. In this case, a promising result means that the particle achieved a J value that is considered low compared to other J values obtained previously. It can also mean that the orbital elements have been minimized to a degree that is considered better than what previous simulations achieved. Using this data, the researcher can

manually decide how small the neighborhood bounds around each particle element should be. Continually applying this technique to promising results can speed up the process of finding an optimal final solution based on sub-optimal results. This process is similar to efforts done in other research to use hybrid strategies that involve using sub-optimal solutions to seed other optimization algorithms, except here, the sub-optimal solutions would seed more simulations from the same algorithm. Alternatively, a researcher can automate this process by using a separate algorithm that modifies the bounds by a certain factor based on the J value and its associated particle. This allows one to refine the bounds to many decimal places to such a degree that a highly optimized result can be achieved. This separate algorithm would also rerun as many simulations as needed based on the adjusted bounds it calculates. Automating the PSO algorithm has an obvious benefit of freeing up researcher time as well as finding a result that might take many dozens of runs to get, due to the randomness factor build into PSO. Both manual and automated strategies were used in this thesis. For the much simpler and lower iteration 2-D and 3-D Time-Fixed runs, manual adjusting was used. For the 2-D and 3-D Time-Variable runs, automated adjusting was used.

CHAPTER 4. RESULTS

4.1 PSO Simulation Setup

The PSO algorithm and other associated code used here were implemented in MATLAB. Using MATLAB was deemed sufficient for the purposes of this thesis even though MATLAB's coding language entails more overhead compared to other languages such as C or C++. This is because the objective of this thesis was not dependent on designing and implementing an efficient algorithm or benchmarking the run-time of different approaches to solve the problem. The runtimes encountered during the testing involved here are reasonable with respect to the number of runs involved.

MATLAB uses a scripting language that is intuitive and offers many packaged built-in functions. The most important built-in function used here is the ode45 integrator, which is a variable step-size integrator based on a Runge-Kutta (4,5) formula called the Dormand-Prince pair [39, 40]. The ode45 integrator uses error tolerances to optimize the size of the time steps used. Both the relative and absolute tolerance values used here are 10^{-9} for the 2-D formulations and 10^{-10} for the 3-D formulations. The next important MATLAB feature is the `parfor` loop. This is a parallel for-loop that can execute iterations of independent code parallel to each other based on the number of "workers" a computer can allow. The number of allowable workers is based on the number of cores or threads a CPU has. In the simulations performed in this thesis, a 4-core 4.0 GHz CPU was used, which means that the `parfor` loops used 4 workers. Finally, MATLAB offers various built-in plotting functions and these were used to plot the change in orbital elements over time as well as construct 3-D depictions of the satellite's position as it performs the burn maneuver.

The algorithmic distinction between the 2-D and 3-D formulations of the GVEs affects the structure of the particles used. The particles for the 2-D Time-Fixed formulation will have the form:

$$\boldsymbol{P}(i) = [\delta_0 \quad \delta_1 \quad \delta_2 \quad \delta_3 \quad E] \tag{36}$$

while the particles for the 2-D Time-Variable formulation have the form:

$$\boldsymbol{P}(i) = [\delta_0 \quad \delta_1 \quad \delta_2 \quad \delta_3 \quad E \quad \Delta t] \tag{37}$$

The particles for the 3-D Time-Fixed formulation will have the form:

$$\boldsymbol{P}(i) = [\delta_0 \quad \delta_1 \quad \delta_2 \quad \delta_3 \quad \varphi_0 \quad \varphi_1 \quad \varphi_2 \quad \varphi_3 \quad E \quad \Delta t] \tag{38}$$

and the particles for the 3-D Time-Variable formulation will have the form:

$$\boldsymbol{P}(i) = [\delta_0 \quad \delta_1 \quad \delta_2 \quad \delta_3 \quad \varphi_0 \quad \varphi_1 \quad \varphi_2 \quad \varphi_3 \quad E \quad \Delta t] \tag{39}$$

All particles use the eccentric anomaly to identify the starting point of the burn maneuver. This choice was based on convenience since eccentric anomaly can easily be converted to true anomaly using Equation (1) and the starting time can be calculated from the eccentric anomaly using the Kepler Equation. The 3-D Time-Fixed version of the particle includes a particle element for the burn period even though the burn period was intended to be user-fixed for Time-Fixed versions. This burn period element was only used as part of a component in the objective function J as follows:

$$C_R = f_R - \left(\frac{n_0}{c}\right)\Delta t_R \tag{40}$$

This was added because it was found during testing that including the fixed burn time in this way greatly increased the speed of 3-D Time-Fixed runs.

The particle elements for the 2-D formulations have the following position bounds, which are related to the orbital plane angle provided in Equation (15):

$$\begin{gathered} -8 \le \delta_0 \le 8 \quad -8 \le \delta_1 \le 8 \quad -8 \le \delta_2 \le 8 \quad -8 \le \delta_3 \le 8 \quad 0 \le E \le 2\pi \\ 0.78 \le \Delta t \le 0.88 \end{gathered} \tag{41}$$

The particle elements for the 3-D formulations have the following position bounds, which are related to the orbital plane and orbit normal angles provided in Eqns. (15) and (16):

$$-22 \le \delta_0 \le 22 \qquad -22 \le \delta_1 \le 22 \qquad -22 \le \delta_2 \le 22 \qquad -22 \le \delta_3 \le 22$$

$$-1 \le \varphi_0 \le 1 \qquad -1 \le \varphi_1 \le 1 \qquad -1 \le \varphi_2 \le 1 \qquad -1 \le \varphi_3 \le 1 \tag{42}$$

$$0 \le E \le 2\pi \qquad 0.72 \le \Delta t \le 0.88 \text{ (Time-Variable)} \qquad 0 \le \Delta t \le \Delta t_R \text{ (Time-Fixed)}$$

The units used in these simulations are normalized to canonical units, a system of units that uses DU as the distance metric and TU as the time metric. Here, 1 DU = radius of the Earth, and 1 TU was chosen so that $\mu = 1\ \text{DU}^3/\text{TU}^2$. The units of the burn period are in TU, the units of the eccentric anomaly to be determined are in radians, and the coefficients of the thruster equations are dimensionless. The initial thrust-to-mass ratio n_0 was set to 0.2 DU/TU^2 and the effective exhaust velocity c was set to 0.5 DU/TU. The burn period for the Time-Fixed cases was set to 0.8 TU, a time period surmised to be a reasonable maximum burn period for a satellite. The number of particles and number of iterations used for each formulation is given below:

	2-D		3-D	
	Time-Variable	Time-Fixed	Time-Variable	Time-Fixed
Num. of particles	60	50	100	100
Num. of iterations	600	500	2000	2000

Table 4.1: Number of Particles and Iterations Used for Different Problem Formulations

The weight values and thresholds used for all cases are summarized in Table 4.2 and Table 4.3. These values were finalized through observation of the simulations which showed that these values led to lower J values and faster convergences.

J Function Component C_h	2-D Time-Fixed		2-D Time-Variable	
	T_h	W_h	T_h	W_h
$\Delta a = a_f - a_0$	0.01	10	1 x 10^{-8}	10
$\Delta e = e_f - e_0$	0.001	10	1 x 10^{-8}	5
$\Delta \omega = \omega_f - \omega_0$	0.01	10	1 x 10^{-8}	10

Table 4.2: Thresholds and Weights for 2-D PSO Formulations

J Function Component C_h	3-D Time-Fixed		3-D Time-Variable	
	T_h	W_h	T_h	W_h
$\Delta a = a_f - a_0$	1.0 x 10^{-9}	5	1.0 x 10^{-9}	10
$\Delta e = e_f - e_0$	1.0 x 10^{-9}	5	1.0 x 10^{-9}	10
$\Delta in = in_f - in_0$	1.0 x 10^{-9}	5	1.0 x 10^{-9}	10
$\Delta \Omega = \Omega_f - \Omega_0$	1.0 x 10^{-9}	5	1.0 x 10^{-9}	10
$\Delta \omega = \omega_f - \omega_0$	1.0 x 10^{-9}	5	1.0 x 10^{-9}	10
C_R	1	10		

Table 4.3: Thresholds and Weights for 3-D PSO Formulations

For convenience, the J equations for the simulation case are stated below. For both 2-D Time-Variable and Time-Fixed formulations, the J function is:

$$J = w_1 \Delta a + w_2 \Delta e + w_3 \Delta \omega \tag{43}$$

For the 3-D Time-Variable simulations, the J function used was:

$$J = w_1\Delta a + w_2\Delta e + w_3\Delta in + w_4\Delta\Omega + w_5\Delta\omega \quad (44)$$

For the 3-D Time-Fixed simulations, the J function used was:

$$J = w_1\Delta a + w_2\Delta e + w_3\Delta in + w_4\Delta\Omega + w_5\Delta\omega + w_6 C_R \quad (45)$$

In this thesis, the "goodness" of a solution is primarily determined by the order of the objective function value because the objective function value is usually an indication to the level of minimization of the function's components. However, a solution that produces a J function that appears not viable can still be considered viable if certain J function components are minimized to the requirements set by the satellite stakeholders.

4.2 Solutions Obtained

The research performed in this thesis uses both 2-D and 3-D formulations of the GVE to attempt to find an optimal solution to some orbit cases. Finding a good solution using either formulation achieves the goal of this thesis. For each case, solutions using Time-Fixed and Time-Variable methods are presented. The cases tested are listed below:

Orbit	a (DU)	e	in (rad)	Ω (rad)	ω (rad)
1	1.22222	0.1	0.6	0.6	0.6
2	2.22222	0.1	0.6	0.6	0.6

Table 4.4: Orbit Cases Tested

Orbit 1 represents an orbit in the upper LEO range since it is near the innermost layer of the Van-Allen radiation belts. Orbit 2 represents a MEO orbit just beyond the inner Van-Allen belt. Because the close proximity to Van-Allen belts would increase the chance of a satellite becoming impaired, these orbits were selected as reasonable candidates for the simulations run. In addition, LEO and MEO orbits were chosen because with existing technology, retrieval or repair operations on satellites are more feasible for LEO and MEO satellites rather than for GEO satellites. The feature that characterizes these orbits as LEO or MEO is

the semi-major axis length. The satellite being simulated is assumed to have 25% of its total mass as structural mass and 75% of its mass as propellant mass. This satellite in Orbit 1 has a perigee of about 637 km above Earth's surface while in Orbit 2, it has a perigee of about 6378 above Earth's surface. The other orbital elements were the same for both the orbits to facilitate easier comparison of results and because the semi-major axis was decided to be the more important orbital element to distinguish different test cases.

The objective function results for these cases are presented below for Time-Variable and Time-Fixed simulations.

	J Values, Time-Fixed			J Values, Time-Variable		
Orbit	Normal Burn	Remainder Burn	Form of GVEs used	Normal Burn	Remainder Burn	Form of GVEs used
1	$< 10^{-25}$	3.838×10^{-12}	2-D	3.544×10^{-12}	5.454×10^{-12}	3-D
2	2.458×10^{-16}	1.110×10^{-16}	2-D	4.645×10^{-13}	$< 10^{-25}$	2-D

Table 4.5: Best J Function Values Obtained for Orbit Cases for Normal and Remainder Burns

The structure of the objective functions uses the absolute value of the difference between the initial and final orbital elements. This implies that small differences would result in a small objective function value for a given particle. For this thesis, solutions that result in J values on or below the order of 10^{-12} are considered highly optimized solutions. The highly optimized solutions for the orbit cases are shown in the table above along with the formulation that achieved such the solution. It is evident from the table that the 2-D formulation of the GVEs tended to provide the best solutions in both Time-Variable and Time-Fixed cases, with the 3-D formulation achieving a highly optimal solution in only one case. This was expected due to the nature of the formulations. The 2-D version ignores orbit-normal components of the GVEs which means it needs to minimize three orbital elements rather than the five that the 3-D version needs. Therefore, the 2-D version's particles require fewer elements and this makes the search space of the solution smaller,

which increases the ease and speed of finding the best solution. Regardless, Table 4.5 shows that the PSO algorithm was successful in finding optimal solutions for these two orbits for both types of burns. This is further supported by the differences in orbital elements achieved by the optimal solutions.

The remainder of this section will use the results for the 3-D Time-Variable formulation of Orbit 1 to illustrate the capability of PSO to solve this minimization problem. The table below displays the changes in orbital elements for this solution as well as additional pertinent solution information.

Orbit 1: 3-D Time-Variable	Normal Burn	Remainder Burn
J Function Value	3.544×10^{-12}	5.454×10^{-12}
Δa (DU)	3.137×10^{-12}	3.130×10^{-14}
Δe	1.840×10^{-13}	5.1734×10^{-12}
Δin (rad)	2.250×10^{-14}	1.78×10^{-14}
$\Delta\Omega$ (rad)	1.007×10^{-13}	6.96×10^{-14}
$\Delta\omega$ (rad)	9.990×10^{-14}	1.622×10^{-13}
E (rad)	2.355	5.790
Δt (TU)	0.766	0.344

Table 4.6: J Function Components and Important Solution Information for Orbit 1 3-D Time-Variable Solution

As the above table shows, the differences in orbital elements for the solution for this orbit are on the order of 10^{-12} or less. For the normal burn, the value of the change in semi-major axis, Δa, is 3.137×10^{-12} DU, which translates to a difference of only 0.02 mm from the original orbit's semi-major axis. Similarly, for the remainder burn, the change in Δa equates

to a difference of around 0.03 mm from the original semi-major axis. The differences in the other orbital elements are small enough to be negligible. Therefore, this solution is highly optimized for this orbit.

As mentioned previously, the solutions obtained here allow the orbital elements to change deviate from the original values as long as the final values are close to the original. The orbital elements over time is referred to in this thesis as the time variance and these time variances are plotted below for the solution presented in Table 4.6. For the normal burn, the time variance plots are shown in Figures 4.1 through 4.5. As can be observed from these plots, each orbital element deviates from its starting value during the burn and is returned to its original value at the end of the burn under control of the PSO algorithm.

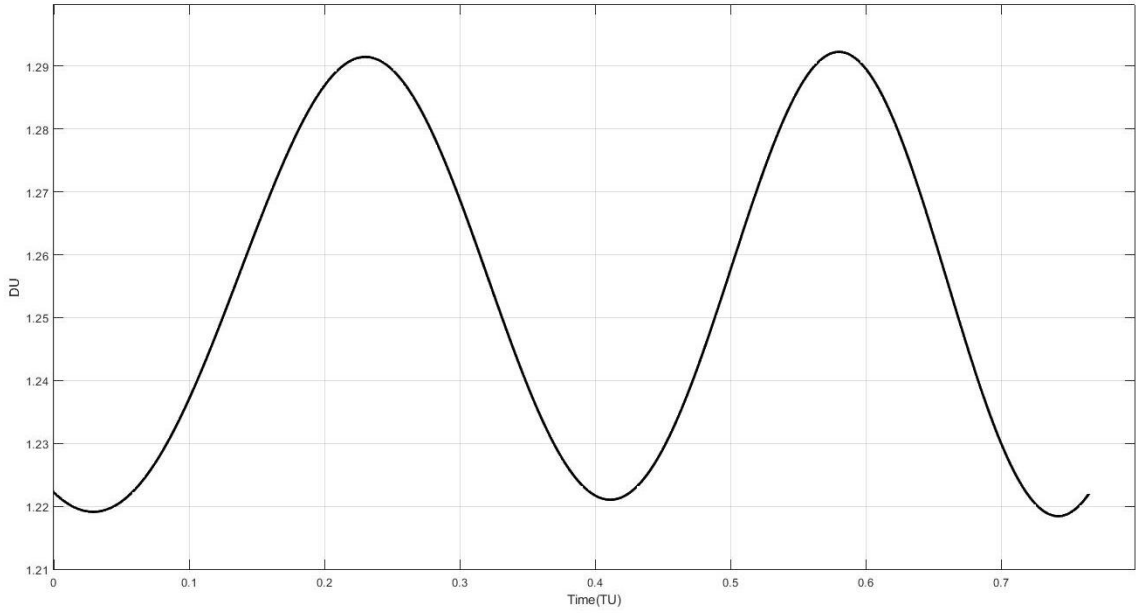

Figure 4.1: Time Variance of Semi-major Axis for Orbit 1 Normal Burn

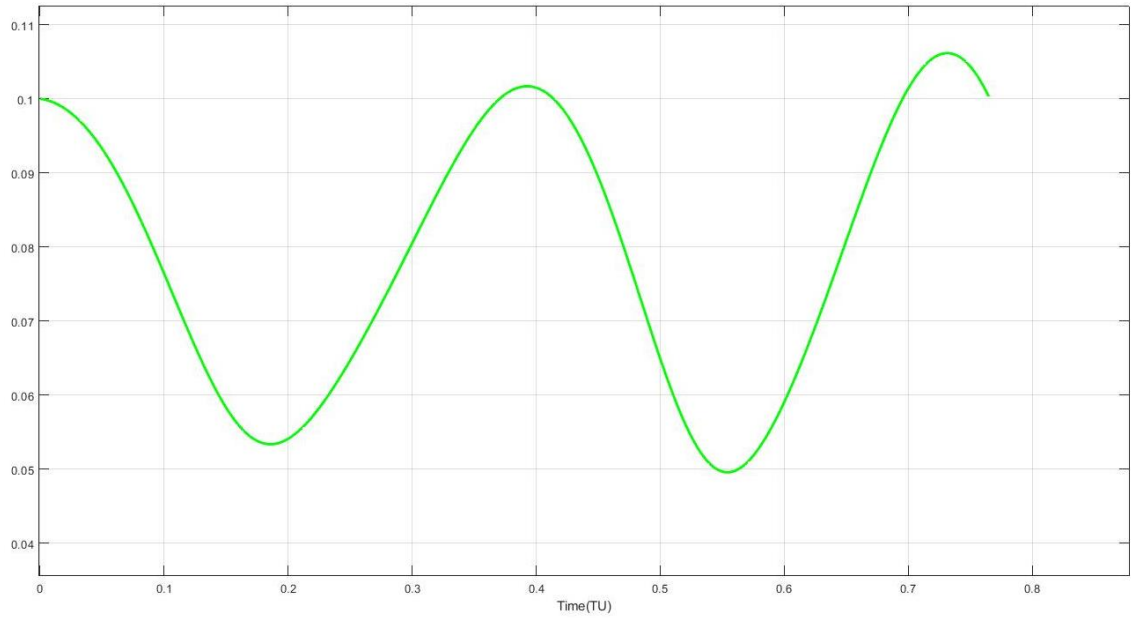

Figure 4.2: Time Variance of Eccentricity for Orbit 1 Normal Burn

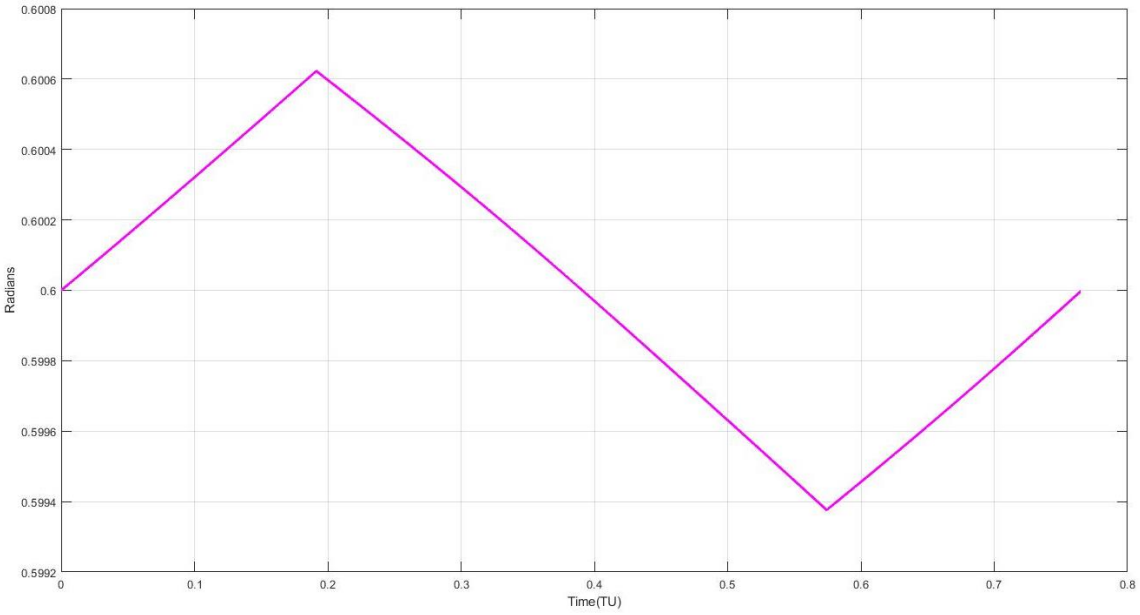

Figure 4.3: Time Variance of Inclination for Orbit 1 Normal Burn

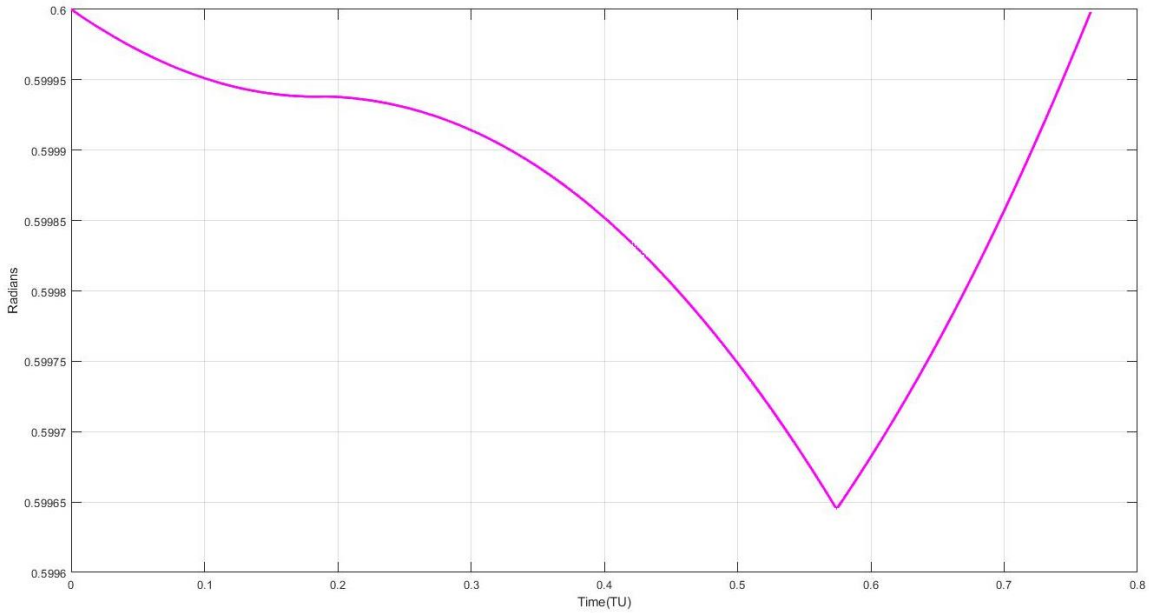

Figure 4.4: Time Variance of RAAN for Orbit 1 Normal Burn

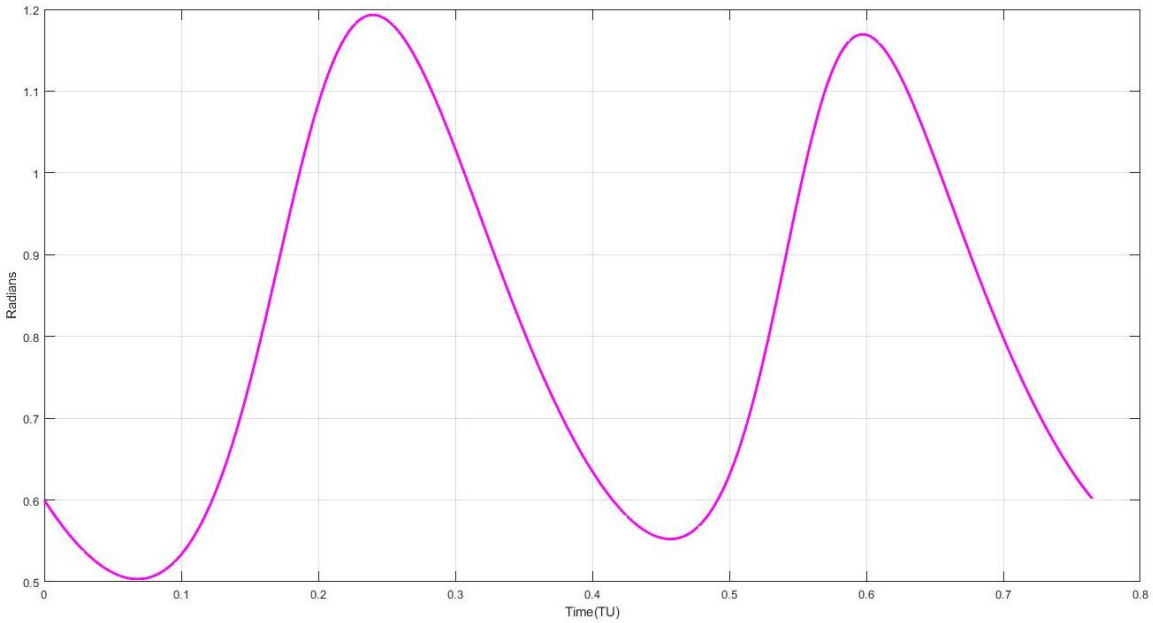

Figure 4.5: Time Variance of Argument of Periapsis for Orbit 1 Normal Burn

For the remainder burn of Orbit 1 in Table 4.6, the plots are shown in Figures 4.6 through 4.10. As before, at the end of the burn, the orbital elements are restored back to the original values they had at the start of the remainder burn.

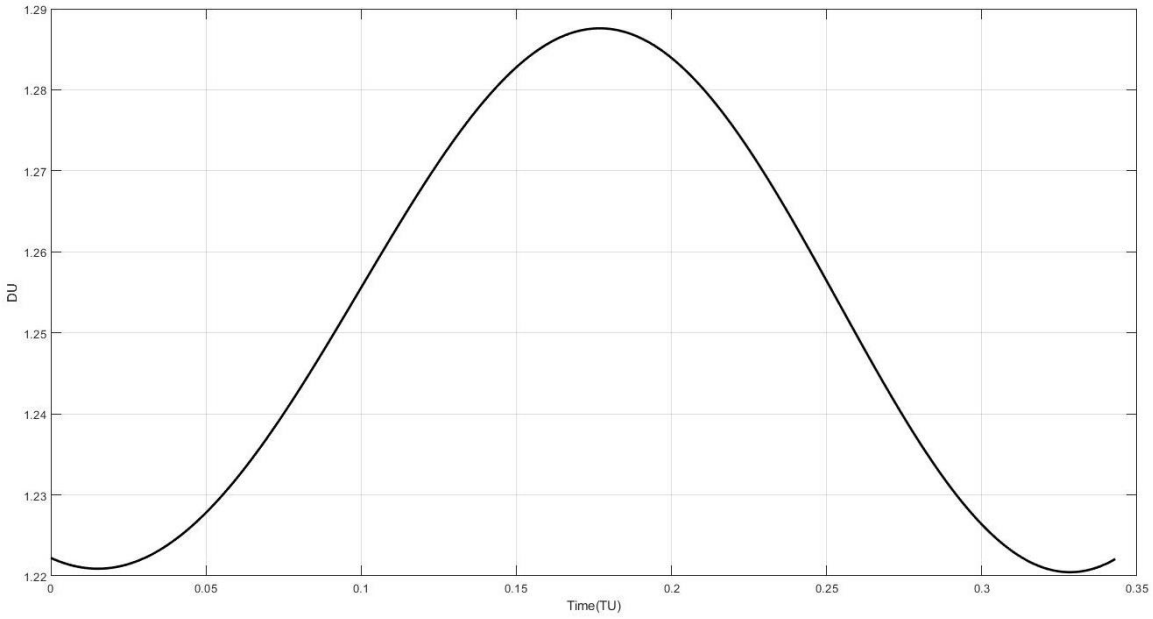

Figure 4.6: Time Variance of Semi-major Axis for Orbit 1 Remainder Burn

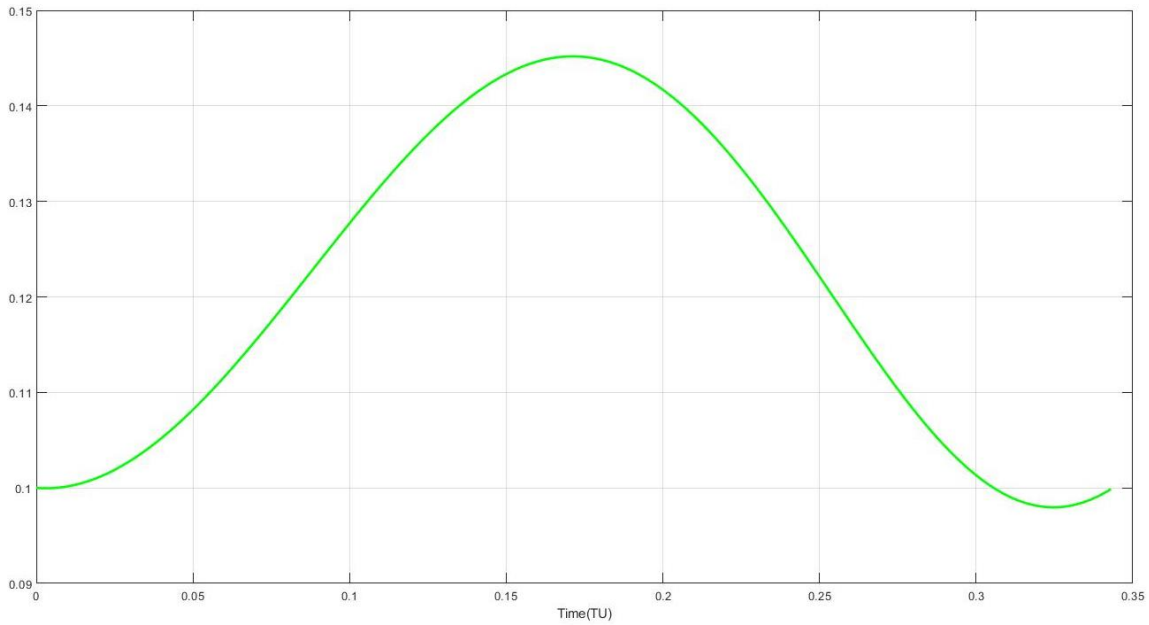

Figure 4.7: Time Variance of Eccentricity for Orbit 1 Remainder Burn

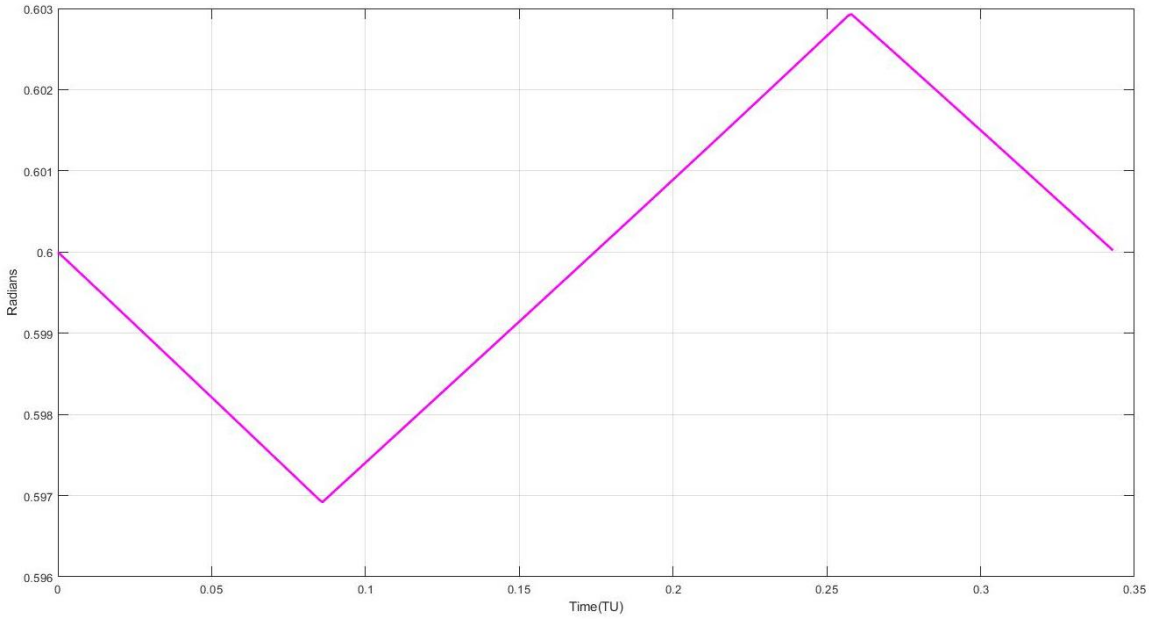

Figure 4.8: Time Variance of Inclination for Orbit 1 Remainder Burn

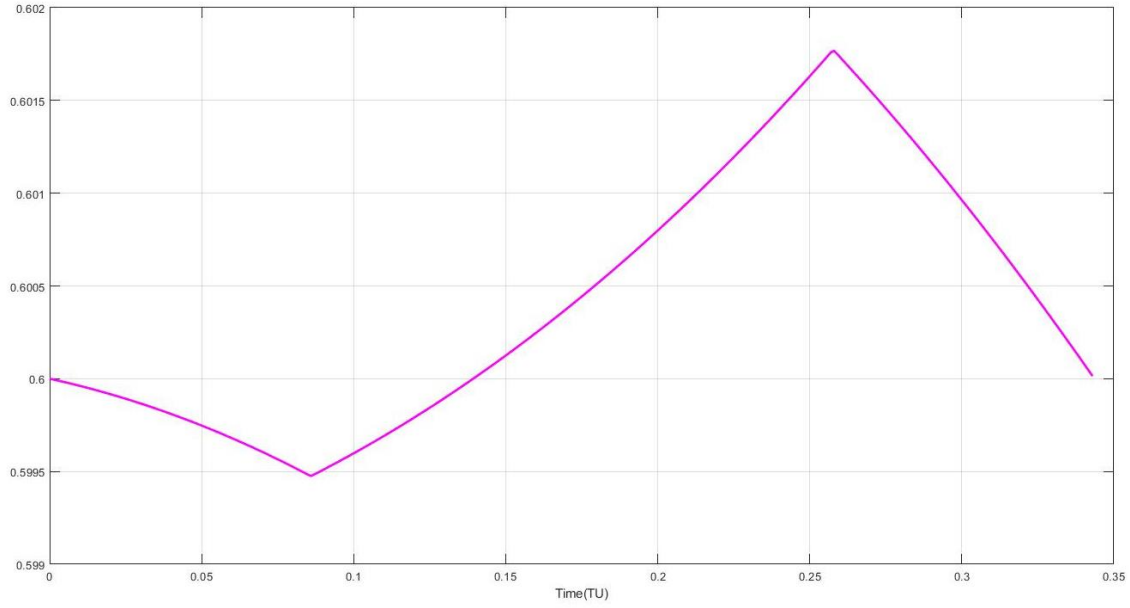

Figure 4.9: Time Variance of RAAN for Orbit 1 Remainder Burn

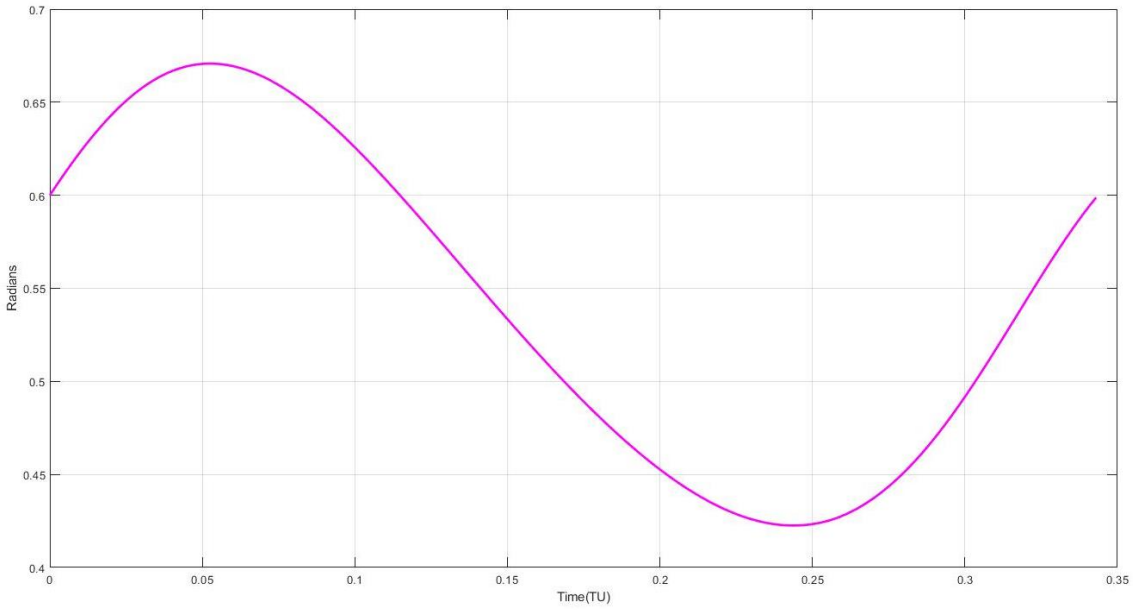

Figure 4.10: Time Variance of Argument of Periapsis for Orbit 1 Remainder Burn

These graphs show how much the orbital elements deviate from the original orbital elements over the course of the normal burn and remainder burn. The plots here are useful for ground control and satellite operators because it shows approximately what the orbital element value is at any given time within the burn period. This means that if the satellite is unable to complete its full burn due to an unforeseen event, the orbital elements of the satellite can still be estimated by using these plots along with the knowledge of when the event occurred within the burn period. This saves time and resources that would have been spent determining the satellite's new orbital elements. Of course, one can also use the PSO solution directly to determine more accurately what the new orbital elements could be at any given time. Additionally, by visualizing the time variances of a solution, the satellite operator/owner may choose to pursue a different solution if the time variances plotted are not deemed acceptable.

For this solution, the inclination and RAAN vary only slightly compared to the semi-major axis, eccentricity, or argument of periapsis. This is because the equation for the orbit-normal thrust-angle φ was found to vary φ by only a few degrees. The inclination and RAAN are dependent on the orbit-normal component of the acceleration vector, which means a

smaller component results in a smaller time rate of change. The other orbital elements have larger ranges of values because the orbital plane angle δ changes drastically over the burn period for both normal burn and remainder burn. This can be seen in the following plots for normal burn and remainder burn:

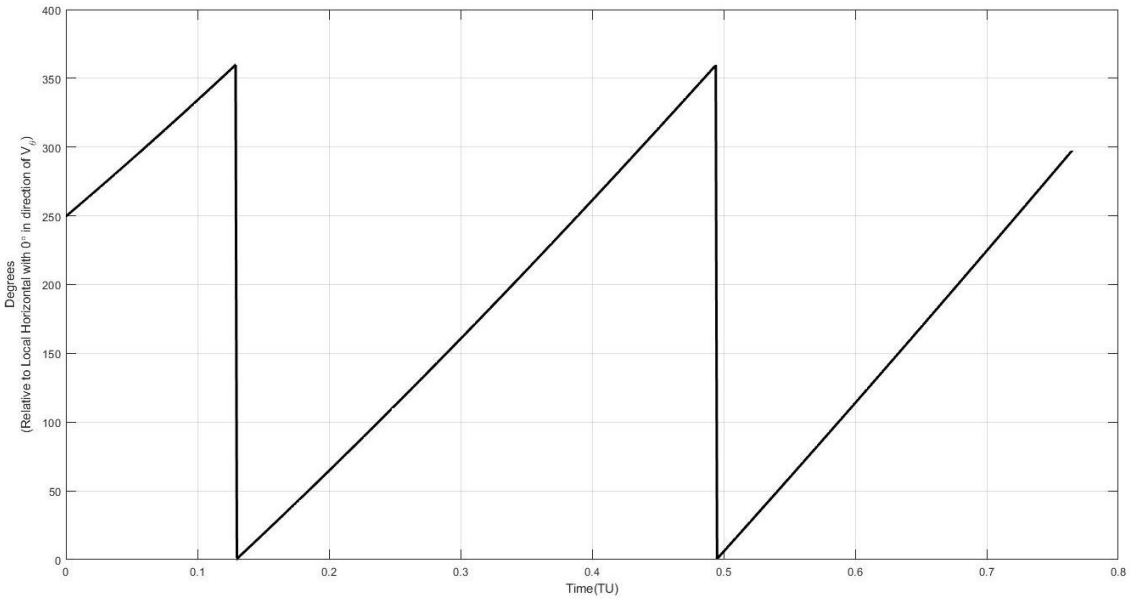

Figure 4.11: Time Variance of Orbit Plane Thrust-Angle δ for Orbit 1 Normal Burn

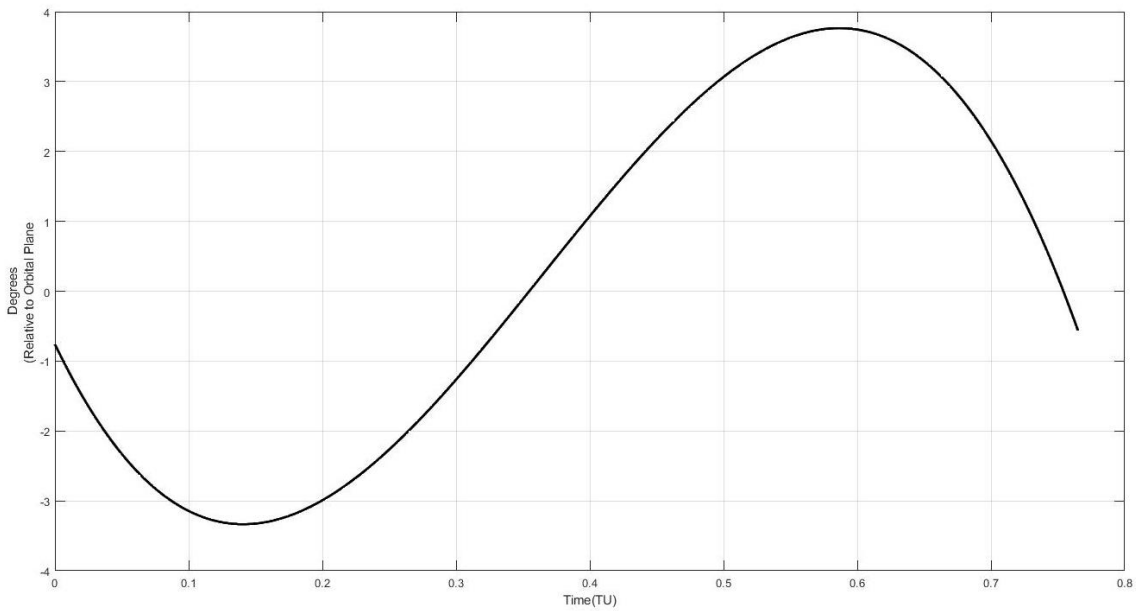

Figure 4.12: Time Variance of Orbit Normal Thrust-Angle φ for Orbit 1 Normal Burn

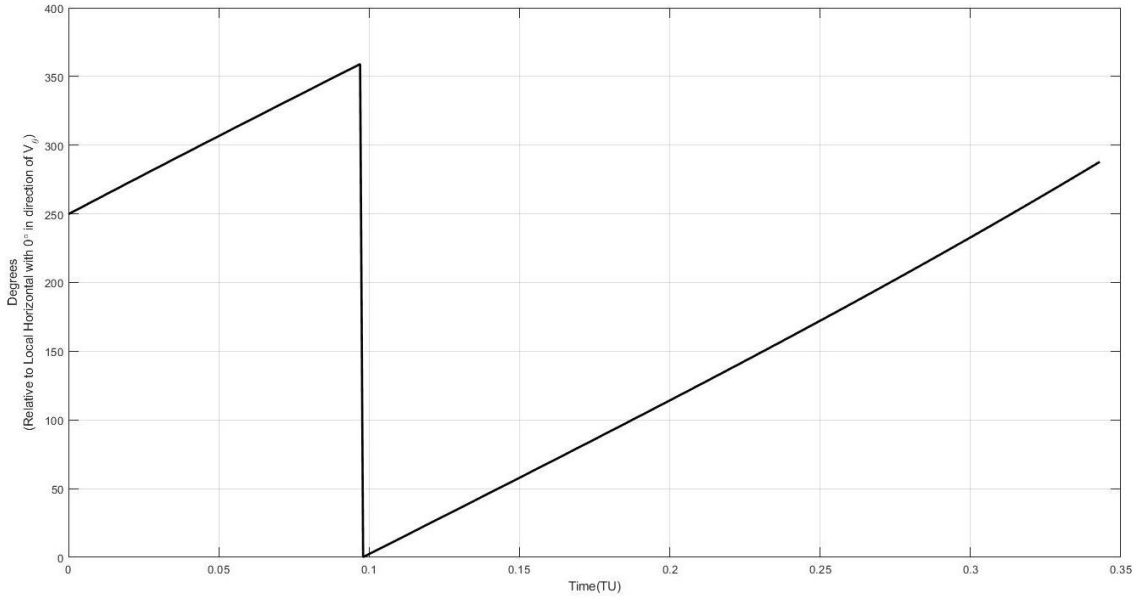

Figure 4.13: Time Variance of Orbit Plane Thrust-Angle δ for Orbit 1 Remainder Burn

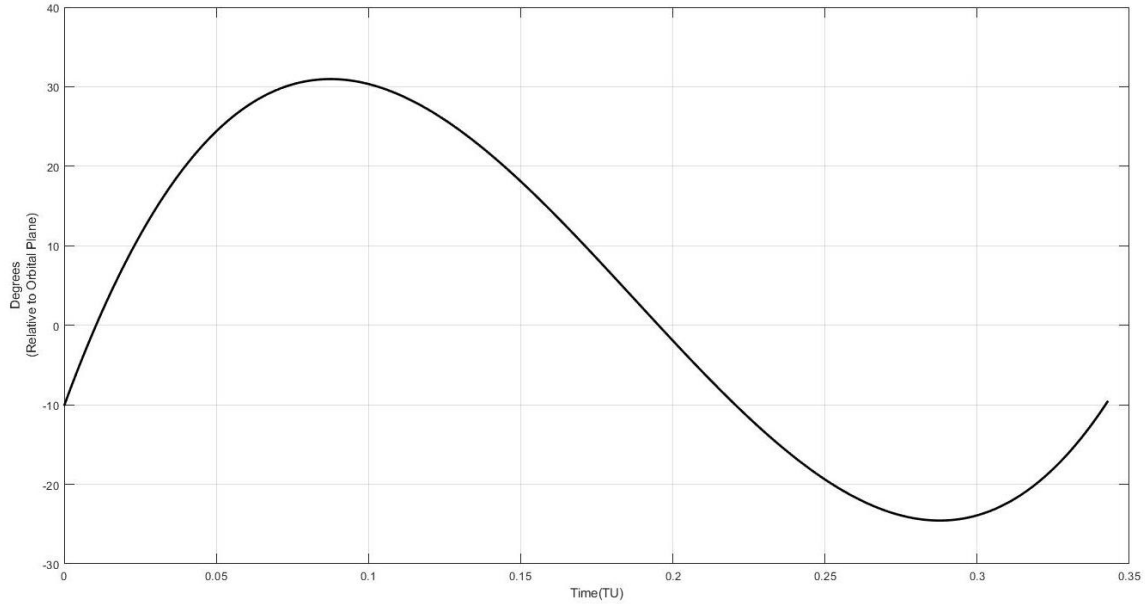

Figure 4.14: Time Variance of Orbit Normal Thrust-Angle φ for Orbit 1 Remainder Burn

The orbit normal thrust-angle for the normal burn and remainder burn vary by about 7° degrees and 55° degrees respectively. On the other hand, the orbital plane thrust-angle changes by over 720° degrees for the normal burn and about 360° degrees for the remainder burn. Satellites thrusters are usually limited to point in a certain angle range with respect to the body of the satellite. In order to point the thrusters in a complete circle over the burn period with respect to the transverse plane of the orbit, the satellite will have to spin about the axis normal to the orbital plane. Modern satellites can achieve this by using the angular momentum wheels located inside the satellite. These momentum wheels can allow the satellite to rotate to continuously point the thrusters as required by the PSO solution. However, satellites that do not have momentum wheels or satellites that require propellant to make attitude adjustments cannot perform this kind of burn maneuver required by this solution.

The thrust directions over the burn periods can also be visualized as shown in Figures 4.15 to 4.18:

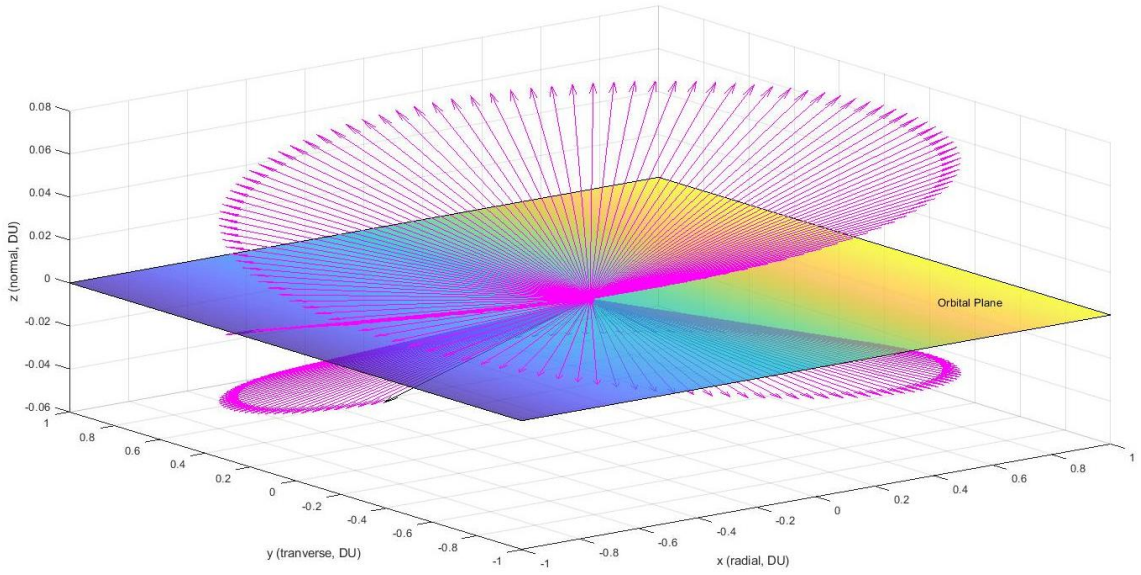

Figure 4.15: Thrust Direction over Time with respect to Orbital Plane for Orbit 1 Normal Burn

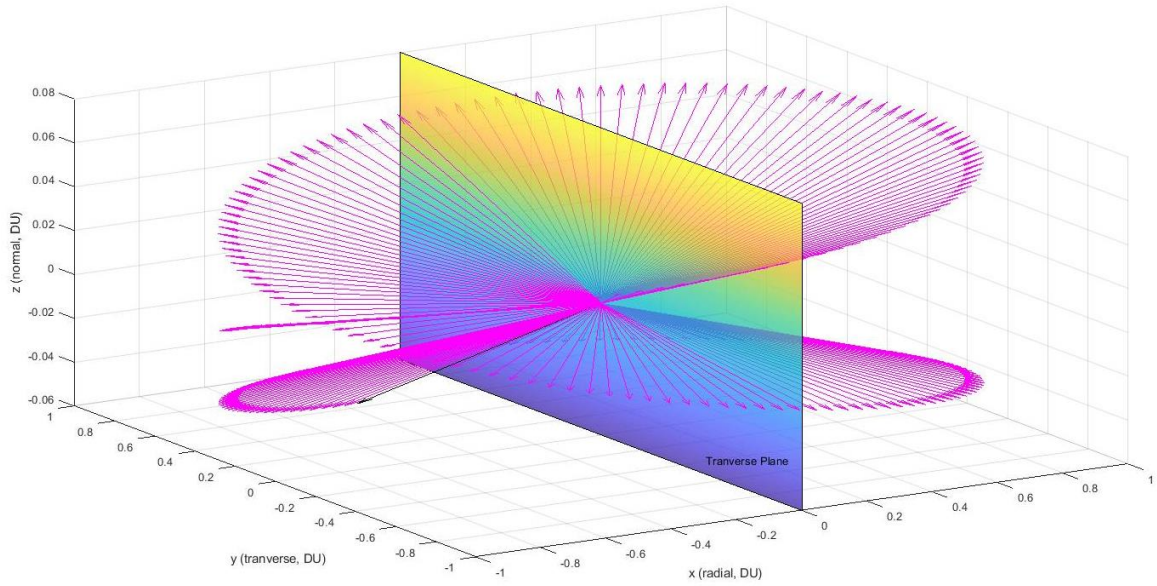

Figure 4.16: Thrust Direction over Time with respect to Transverse Plane for Orbit 1 Normal Burn

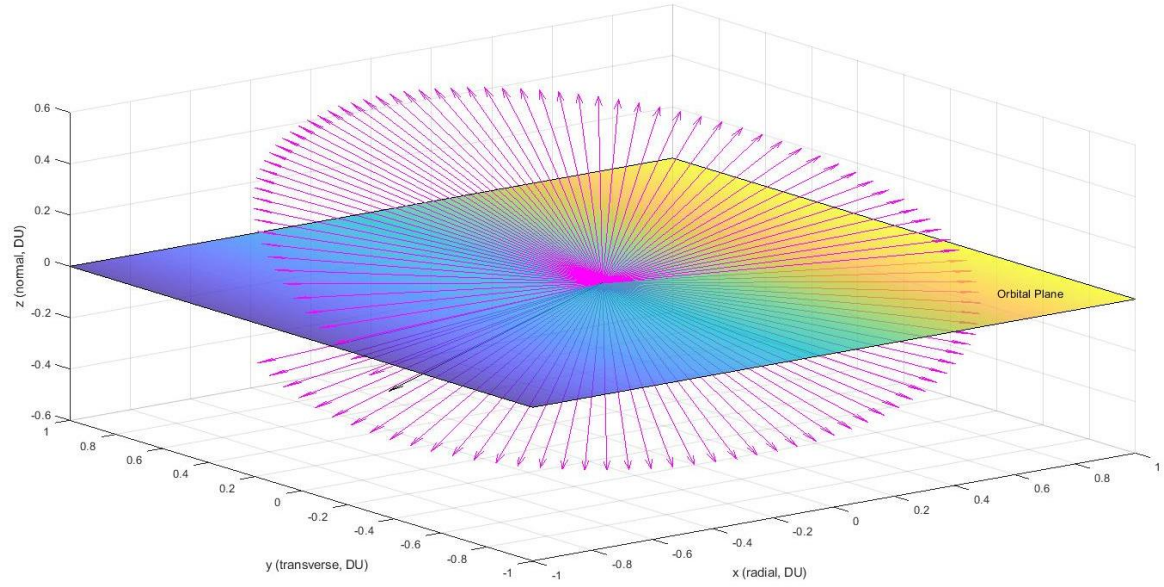

Figure 4.17: Thrust Direction over Time with respect to Orbital Plane for Orbit 1 Remainder Burn

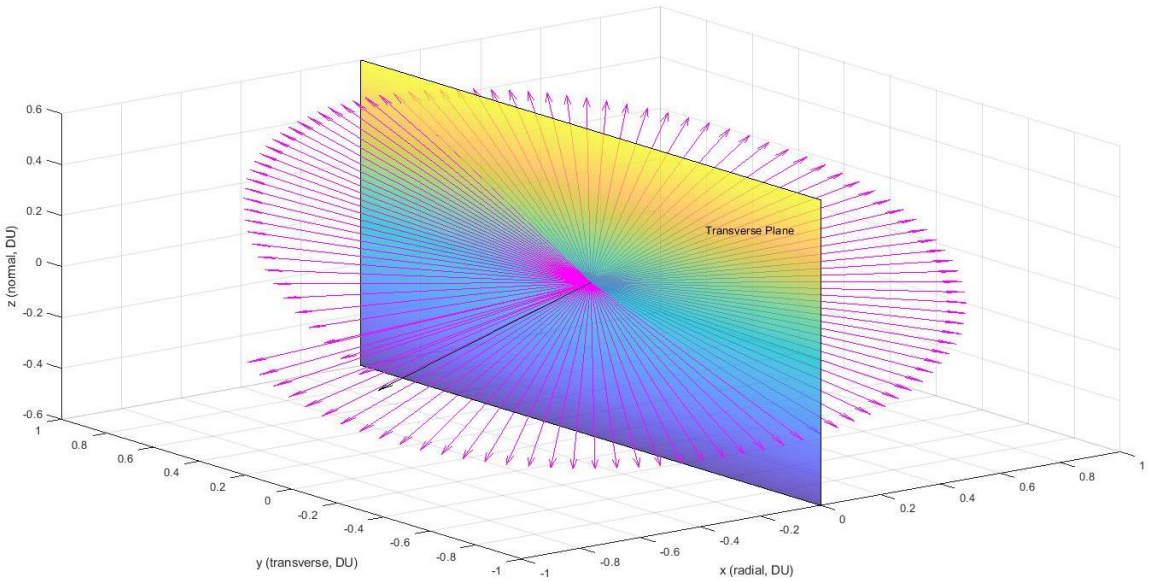

Figure 4.18: Thrust Direction over Time with respect to Transverse Plane for Orbit 1 Remainder Burn

In the above plots, the black arrow shows the starting direction of the thrust vector and by following the pink arrows, one can intuitively see how the thrust vectors change over the particular burn period. This visualization allows an operator to understand how far above or below the thrusters need to point and how much the satellite should rotate to allow the thrusters point in a complete arc.

While the PSO solution was calculated with respect with the orbital plane, the path of the satellite during the burn maneuver can be calculated based on the orbital element time variances. The plots below show path taken by the satellite with respect to the path of the original orbit.

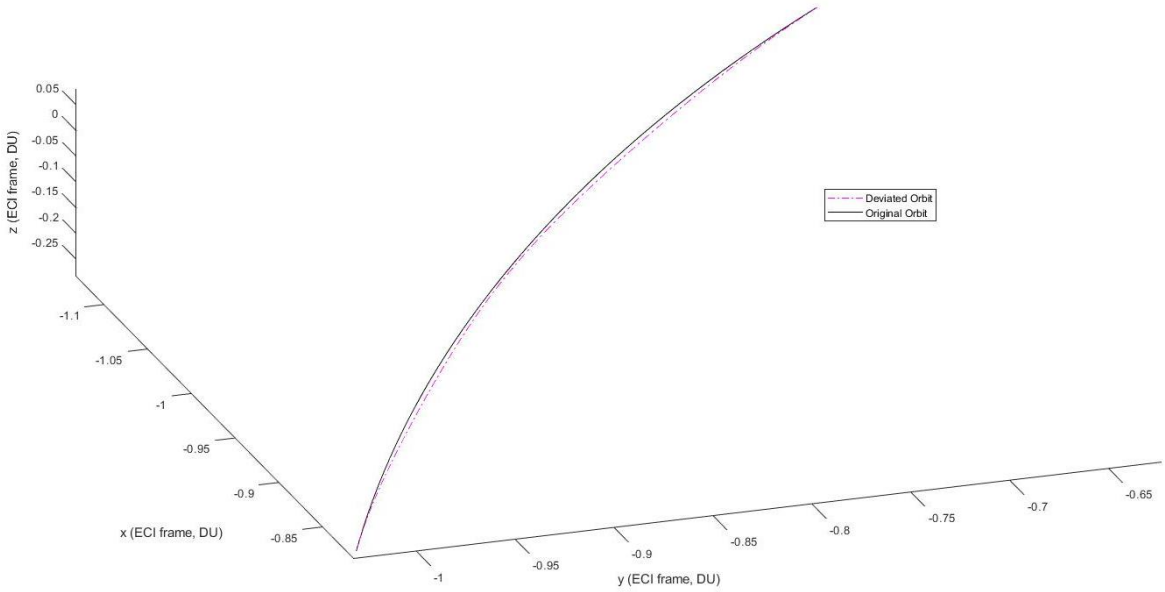

Figure 4.19: Satellite Path (pink) with Respect to Original Orbit (black) for Orbit 1 Normal Burn

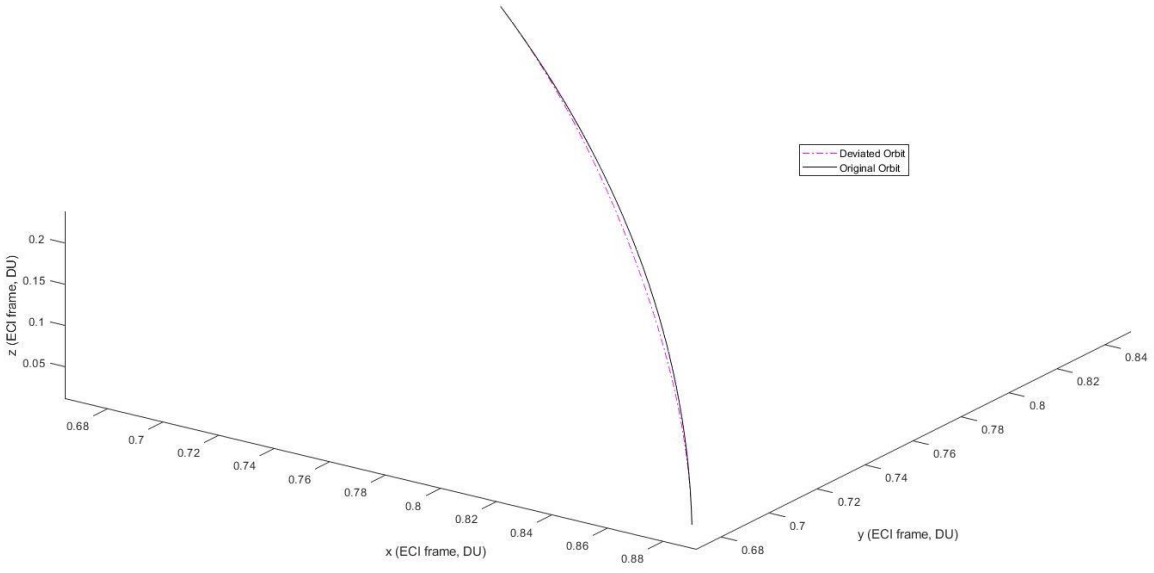

Figure 4.20: Satellite Path (pink) with Respect to Original Orbit (black) for Orbit 1 Remainder Burn

These three-dimensional plots are in the ECI frame and show that the satellite's path during each of the burns is relatively close to the original orbit. Upon closer inspection, it is evident that at the start of the maneuver, the satellite's path splits from the original path, and at the end point of the burn maneuver, the satellite's deviated path (pink) rejoins the path defined by the original orbit. The following graphs offer a better visualization with respect to the Earth where on Orbit 1 the satellite would start its normal burn and remainder burn maneuvers.

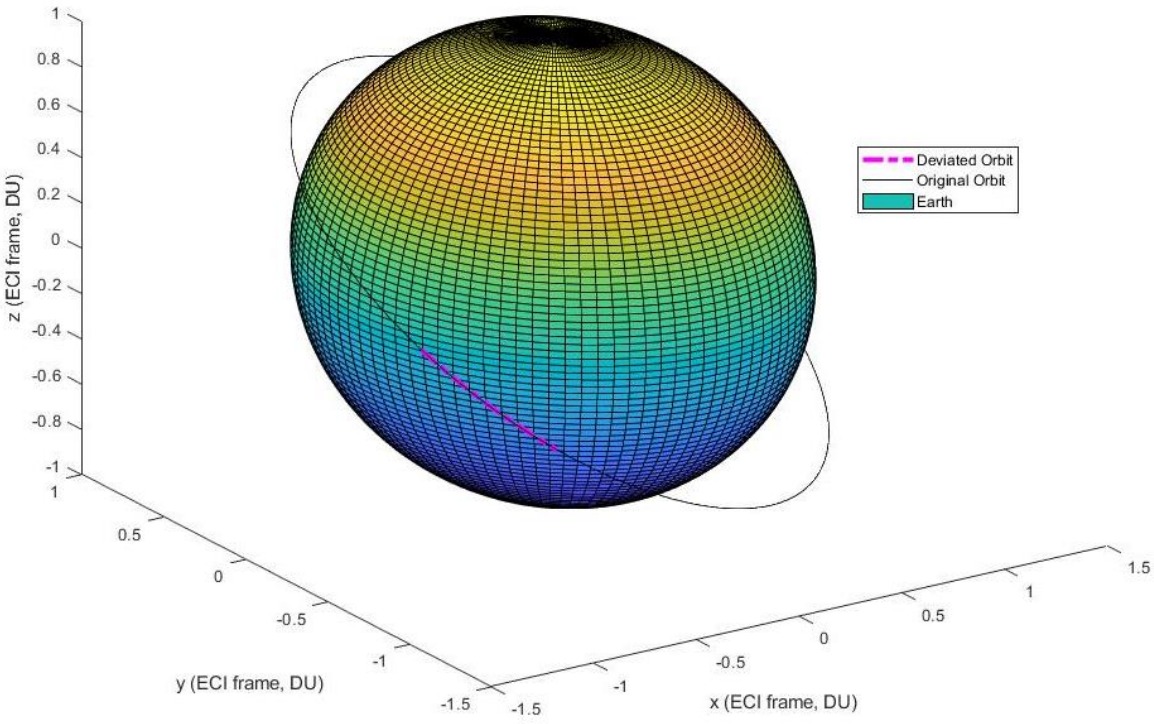

Figure 4.21: Satellite Deviated Path (pink) and Original Orbit with Respect the Earth for Orbit 1 Normal Burn

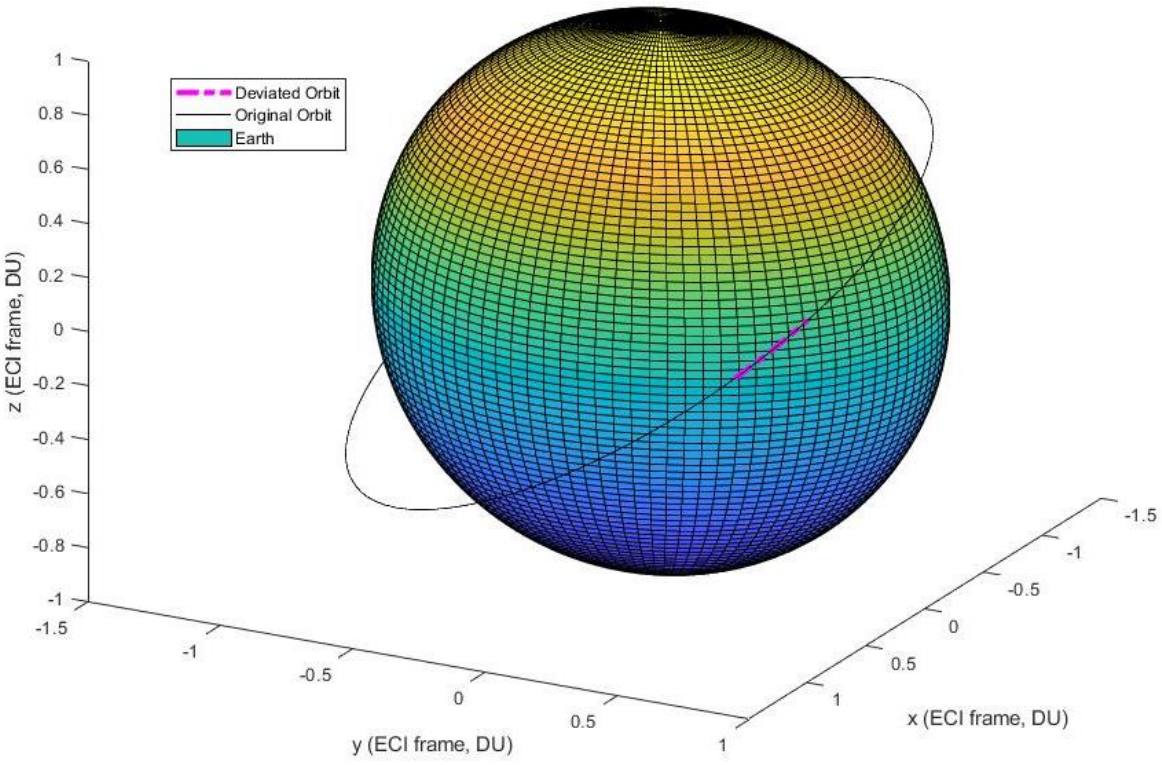

Figure 4.22: Satellite Deviated Path (pink) and Original Orbit with Respect the Earth for Orbit 1 Remainder Burn

The data and plots here show that the PSO algorithm employed was successfully able to attain optimized solutions for some sample orbits. In particular, it was able to find a highly optimized solution for one of the orbits, lending credence to the effectiveness of PSO towards solving this minimization problem.

4.3 Discussion and Observations about PSO

As expected, the PSO algorithm was relatively straightforward to implement and test for both 2-D and 3-D formulations. In addition, the simple structure of the algorithm was helpful for debugging purposes and adding minor updates or changes to refine the results. PSO also worked well with the integration involved, which supports the notion that the PSO algorithm is generic enough to be applied to a specific problem while still providing the means to obtaining a highly optimized and usable solution. Using MATLAB for this endeavor proved very favorable because it allowed for quick implementation and easy visualization of results.

One observation made during testing was that the convergence speed was dependent on three factors: the number of particles, the number of iterations, and the structure of the particle. During each iteration, numerical integration using the ode45 tool is performed on each particle. Therefore, one can intuitively understand why using more particles and more iterations would increase the run time of a simulation. More particles mean that more integrations are performed over one iteration, and more iterations means that this set of integrations is repeated linearly. However, the most important factor to convergence speed in this minimization problem was the particle structure. The simulations were run for both Time-Fixed and Time-Variable formulations, which means that the particles would either have the burn time period fixed by the user, or have the burn time period determined by the PSO algorithm. It was found during testing that keeping the time period user-fixed resulted in considerably slower simulations compared to the simulations that used PSO-determined time periods. The Time-Fixed iterations performed each iteration at a rate of 2-3 seconds per iteration and in some runs at a rate of 4 seconds per iteration. The Time-Variable had much better iteration rates at less than one second per iteration. This meant that Time-Variable runs would complete in about 5 to 20 minutes, while the Time-Fixed runs would complete about 30 minutes for the 2-D version and over an hour for the 3-D version. To save time, the 3-D Time-Fixed version of the algorithm was updated to have particle structure similar to the 3-D Time-Variable version, as shown in Equation (39). In this case, the algorithm would not determine the time period since it was known, but was modified to instead to forcibly use the known time period by including the fuel minimization term in Equation (40). It was deemed unnecessary to perform the same update to the 2-D Time-Fixed because this version uses fewer particles and fewer iterations already.

Another observation made was the tendency of the particles to stagnate on a particular local solution. Stagnation in a PSO context refers to the objective function values remaining the same or negligibly changing as iterations increase. The behavior observed is that the global best J value would decrease for many iterations and then remain level for a long period of iterations, sometimes even for the remainder of the simulation. Because the J function is a minimization function, this means that stagnated particles are all “trapped” in a local minimum. This was especially detrimental to the 3-D formulations because the iterations were set to 2000 to allow for better convergence results. Stagnation in 3-D

simulations meant that the global J value remained level for most of the 2000 iterations. The stagnation encountered is a consequence of both the nature of the search space and the randomness factor built into PSO. To combat this, the algorithm for all formulations except 2-D Time-Fixed were updated to re-randomize half of the particles' positions after the global J value has stagnated for some iteration threshold. This re-randomization would repeat if the J values stagnated again. This technique had limited success, helping in some simulations while having no observable effect in others.

The PSO algorithm's tendency towards possible local optima influenced the minimization methodologies used. This tendency was evident after viewing the results of multiple simulations. It was observed that almost all simulations resulted in final J values on the order of 10^{-1} to 10^{-4}. This indicates that the final orbital elements were significantly different from the initial orbital elements within the context of the orbit's original characteristics. For example, a $\Delta a \approx 1 \times 10^{-4}$ for Orbit 1 would equate to a difference of about 635 meters between the original semi-major axis and final semi-major axis. This was considered an unsatisfactory result for this minimization problem. It was rare to get a solution that achieved a final J value on the order of 10^{-8} or less in one attempt. One strategy used to guide the algorithms towards better results was to add weights to the objective function and adjust them as necessary. Both fixed values and polynomially-derived values were used during initial testing. The testing shows that including weight factors improved the quality of the final solutions. However, increasing the magnitude of the weights usually made the obtained solutions worse. Thus, it was decided to lightly weight the J function components, as shown in Tables 4.2 and 4.3. The most important technique that aided in the minimization process was the adjusting of the particles' position bounds based on the last best sub-optimal solution. Adjusting the bounds in this way after finding a promising solution significantly improved the chances of finding a highly optimized solution. This was especially useful for 3-D versions of the algorithm because the search space was set to be much larger than the search space for the 2-D versions. This strategy enabled the algorithm to find the solutions presented in the previous section. Automating this process was also highly useful as it allowed the PSO algorithm to run hundreds of times without human input to continually narrow down the search space and find the highly optimal solution.

CHAPTER 5. CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

The research conducted here shows that PSO is an effective and simple algorithm for finding optimal solutions to a multivariable problem. Combining PSO concepts with the physical model of the Gauss Variational Equations to solve a specific orbital minimization problem was effective. This is evidenced by the results obtained, which contain solutions that are able to minimize the change in orbital elements while consuming all propellant to at least 10^{-12} order of accuracy. Results such as these are highly satisfactory for the demands of this problem. PSO's shortcomings were easily overcome without modifying the core algorithm by adding several adjustments that would guide the particles of the PSO to more efficiently search for the optimal solution. For the most part, the PSO simulations were run in reasonable times despite needing readjustment of parameters for the particle's position bounds and despite being implemented on an overhead-heavy tool like MATLAB. The reasonable run times and convergence speed was also partly because of the machine running the algorithm, which was able to run 4 parallel loops per iteration at 4 GHz. This means that using a faster CPU with more cores will result in even faster simulation run times. The solutions acquired through the methodology described in this thesis could help satellite operators prepare an impaired satellite for retrieval or repair. In particular, the solutions provide a complete function with respect to time for the thrust-angles. Satellite operators could potentially uplink this information to the satellite and command the satellite to automatically begin and perform the maneuver using the parameters derived from the PSO algorithm.

5.2 Recommendations for Future Research

The focus of this thesis research was on maximizing the satellite propellant consumption prior to a satellite's in-orbit retrieval or repair such that its orbital parameters were unchanged, subject to some assumptions. Future research on this subject could consider solutions to the same problem under other assumptions. For example, the optimized solutions found here rely on the assumption that the impaired satellites use momentum wheels to change the attitude rather than using propellant. For satellites that use propellant for attitude changes, a different PSO formulation is needed that either uses

the attitude adjustment fuel consumption as an additional constraint or uses it as part of the solution to be determined. This type of solution, together with the solutions in this thesis, would be applicable to most impaired satellites currently orbiting the Earth that are amenable for repair or retrieval.

Other future research can explore related orbital solutions using the GVEs. The GVE model used in this thesis could be used to solve a general-purpose orbital transfer problem, in which a satellite must move from one orbit with a set of orbital elements to another orbit with another set of orbital elements while minimizing the propellant used. The setup would be similar to the one outlined in this paper and the thrust-angles could be calculated in a similar manner shown here. Problems of the type presented in this thesis could also be solved by using other PSO algorithm variants mentioned previously in Chapter 3, instead of the original algorithm proposed by Kennedy and Eberhart [34]. The other PSO-inspired algorithms show improved convergence and less stagnation for the problems they solved, and it is possible that these alternative PSO algorithms represent a better fit for a complex GVE-related problem.

REFERENCES

1. Available: http://www.unoosa.org/oosa/osoindex/search-ng.jspx?lf_id= , accessed September 21, 2019.
2. Available: https://www.ucsusa.org/nuclear-weapons/space-weapons/satellite-database , accessed September 21, 2019.
3. Pritchard, W. L., Suyderhoud, H. G., Nelson, R. A., *Satellite Communication Systems Engineering*, 2nd Edition, Chapter 5, Prentice Hall, Englewood Cliffs, NJ.
4. Witze, A. "The Conquest to Conquer Earth's Space Junk Problem.", *Nature*, Issue 561, pp. 24-26, 05 September, 2018. Internet: https://www.nature.com/articles/d41586-018-06170-1, accessed September 14, 2019.
5. Available: http://fp7-spacecast.eu/help/bg_sa.pdf, accessed September 15, 2019.
6. Sanchez, A. H., Soares, T., and Wolahan, A. "Reliability Aspects of Mega-Constellation Satellites and Their Impact on the Space Debris Environment", Annual Reliability and Maintainability Symposium, IEEE Conference Paper, 2017.
7. Wilson, W. "The United States Air Force Acquisition Annual Report", Fiscal Year 2017. Available: www.af.mil/Portals/1/documents/1/annual_report_web.pdf [accessed September 21, 2019]
8. Bedingfield, K. I., Leach, R. D., and Alexander, M. B., "Spacecraft System Failures and Anomalies Attributed to the Natural Space Environment", NASA Reference Publication 1390, NASA Marshall Space Flight Center, Alabama, August 1996.
9. Cantieri, J. "Fixing Satellites in Space", December 14, 2017. [online] URL: http://www.astronomy.com/news/2017/12/fixing-satellites-in-space [accessed September 16, 2019]
10. National Space and Aeronautics Administration. [online] https://sspd.gsfc.nasa.gov/images/Restore_L_Factsheet_092717_G.pdf [accessed September 20, 2019].
11. National Space and Aeronautics Administration. [online] https://sspd.gsfc.nasa.gov/restore-L.html [accessed September 20, 2019].
12. National Space and Aeronautics Administration. Nov 3, 2015. [online] https://www.nasa.gov/image-feature/robotic-eyes-to-assist-satellite-repairs-in-orbit [accessed September 23, 2019].

13. DARPA [online] https://www.darpa.mil/program/robotic-servicing-of-geosynchronous-satellites [accessed September 20, 2019].
14. European Space Agency, "SMA Valves to Prevent In-Orbit Explosions", [online] URL: http://blogs.esa.int/cleanspace/2017/07/21/sma-valves-to-prevent-in-orbit-explosions/ [accessed September 24, 2019].
15. Lacovic, R. F. et al. "Management of Cryogenic Propellants in a Full-Scale Orbiting Space Vehicle", NASA Technical Note, NASA TN D-4571, May 1968.
16. Cichan, T., Melton, R.G., Spencer, D.B. (2001) "Control Laws for Minimum Orbital Change – The Satellite Retrieval Problem", *Journal of Guidance, Control, and Dynamics,* 24(6), pp. 1231-1233.
17. Zahroof, T., Shageer, H., Bylard, A. and Pavone, M., "Perception-Constrained Robot Manipulator Planning for Satellite Servicing", IEEE Aerospace Conference, 2-9 March 2019, Big Sky, Montana, USA.
18. Xu, W., Liang, B., Gao, D. and Xu, Y., "A Space Robotic System Used for On-Orbit Servicing in the Geostationary Orbit", IEEE/RSJ International Conference on Intelligent Robots and Systems, October 18-22, 2010, Taipei, Taiwan.
19. Landzettel, K., Preusche, C., Albu-Schaeffer, A., Reintsema, D., Rebele, B. and Hirzinger, G., "Robotic On-Orbit Servicing – DLR's Experience and Perspective", IEEE/RSJ International Conference on Intelligent Robots and Systems, October 9-15, 2006, Beijing, China.
20. Hou, P., Liu, Y., Xie, Z. and Liu, H. "Development of a Space Arm-Hand System for On-orbit Servicing", International Conference on Mechatronics and Automation, 2-5 August 2015, Beijing, China.
21. Han, Y. and Hong, J. "Retrieval Strategy for Failed Satellite on Tether's Optimal Balance Swing Angle", *Journal of Systems Engineering and Electronics*, Vol. 30, No. 4, pp. 749-759, August 2019.
22. Chunke, L., "Multi-pulse Maneuver Orbit Optimization based on a New Hybrid Algorithm", IEEE International Conference on Intelligent Computing and Intelligent Systems, 29-31 October 2010, Xiamen, China. DOI: 10.1109/ICICISYS.2010.5658447.
23. Shaw, Matthew J. (2018) "Design of Interplanetary Trajectories with Multiple Synergistic Gravitational Assist Maneuvers Via Particle Swarm Optimization", Master's Thesis, The Pennsylvania State Univerisity, May 2018.

24. Pontani, M. and B. A. Conway (2010) “Particle Swarm Optimization Applied to Space Trajectories,” *Journal of Guidance, Control, and Dynamics*, 33, pp. 1429–1441.
25. Vtipil, S. and Warner, J. G., “Earth Observing Satellite Orbit Design vis Particle Swarm Optimization”, AIAA/AIS Astrodynamics Specialists Conference, 4-7 August 2014, San Diego, CA
26. Zhang, Q., Sun, F., Wen, Y. and Chen, J., “On-orbit Servicing Task Allocation for Spacecrafts Using Discrete Particle Swarm Optimization Algorithm”, *Advanced Materials Research,* Vols. 268-270, pp. 574-580, July 4, 2011. DOI:10.4028/www.scientific.net/AMR.268-270.574.
27. Gao, Z, Guo Y., Ma, G. and Feng, Z., “Orbit Maneuver for GEO On-orbit Service Satellite Using Hohman Transfer”, Chinese Automation Congress, 20-22 October 2017, Jinan, China. DOI: 10.1109/CAC.2017.8243382
28. Petropoulos, A. E., “Simple Control Laws for Low-Thrust Orbit Transfers”, AAS/AIAA Astrodynamics Specialists Conference, Paper AAS 03-630, 3-7 August 2003, Big Sky, Montana.
29. Battin, Richard H., *An Introduction to the Mathematics and Methods of Astrodynamics,* Revised Edition*,* Chapter 10, AIAA, Inc., Reston, VA.
30. Spillar, D., Ansalone, L., Curti, F. (2016) “Particle Swarm Optimization for Time-Optimal Spacecraft Reorientation with Keep-Out Cones”, *Journal of Guidance, Control, and Dynamics*, 33(2), pp. 312-325.
31. Melton, R.G., (2014) “Hybrid Methods For Determining Time-Optimal, Constrained Spacecraft Reorientation Maneuvers”, Acta Astronautica, 94(1), pp. 294-301.
32. Muthuswamy, S., Lam, S. (2011) “Discrete Particle Swarm Optimization For The Orienteering Problem”, *International Journal of Industrial Engineering*, 18(2), pp. 92-102.
33. Pan, Q-K., Tasgetiren, M. F., Liang, Y-C. (2008). “A Discrete Particle Swarm Optimization Algorithm for the No-wait Flowshop Scheduling Problem”, *Computers & Operations Research*, 35, pp. 2807–2839.
34. Kennedy, J. and R. Eberhart (1995) “Particle Swarm Optimization,” in Proceedings of ICNN’95 - International Conference on Neural Networks, Vol. 4, pp. 1942–1948.
35. Abido, M. A. (2002) “Optimal Power Flow Using Particle Swarm Optimization”, *Electrical Power Energy Systems*, 24, pp. 563–571.

36. Fukuyama, Y., Takayama, S., Nakanishi, Y., & Yoshida, H. (1999). "A Particle Swarm Optimization for Reactive Power and Voltage Control in Electric Power Systems", Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, FL.
37. Brandstatter, B., Baumgartner, U. (2002). "Particle Swarm Optimization – Mass-Spring System Analogon", *IEEE Transactions on Magnetics*, 38, pp. 997–1000.
38. Pluhacek, M., Senkerik, R., Zelinka, I. (2014) "Gathering Algorithm: A New Concept of PSO-based Metaheuristic with dimensional mutation" in Proceeding of the 2014 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, Florida, USA, 9-12 Dec, pp. 42-47.
39. Dormand, J. R. and P. J. Prince, "A Family of Embedded Runge-Kutta Formulae," *J. Comp. Appl. Math*., Vol. 6, 1980, pp. 19–26.
40. Shampine, L. F. and M. W. Reichelt, "The MATLAB ODE Suite," *SIAM Journal on Scientific Computing*, Vol. 18, 1997, pp. 1–22