9-1-2018

# Optimal Finite Thrust Guidance Methods for Constrained Satellite Proximity Operations Inspection Maneuvers

Eric R. Prince


# OPTIMAL FINITE THRUST GUIDANCE METHODS FOR CONSTRAINED SATELLITE PROXIMITY OPERATIONS INSPECTION MANEUVERS

DISSERTATION

Eric R. Prince, Capt, USAF

AFIT-ENY-DS-18-S-071

**DEPARTMENT OF THE AIR FORCE**
**AIR UNIVERSITY**

***AIR FORCE INSTITUTE OF TECHNOLOGY***

**Wright-Patterson Air Force Base, Ohio**

AFIT-ENY-DS-18-S-071

# OPTIMAL FINITE THRUST GUIDANCE METHODS FOR CONSTRAINED SATELLITE PROXIMITY OPERATIONS INSPECTION MANEUVERS

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Eric R. Prince, BS, MS

Capt, USAF

September 2018

AFIT-ENY-DS-18-S-071

# OPTIMAL FINITE THRUST GUIDANCE METHODS FOR CONSTRAINED SATELLITE PROXIMITY OPERATIONS INSPECTION MANEUVERS

DISSERTATION

Eric R. Prince, BS, MS
Capt, USAF

Committee Membership:

Dr. Richard G. Cobb
Chairman

Capt Joshuah A. Hess, PhD
Member

Dr. Matthew C. Fickus
Member

# Abstract

For an inspector satellite characterizing a resident space object (RSO) in geosynchronous orbit (GEO), time-optimal and fuel-optimal maneuvers are key to improving the capability and mission life of the inspector satellite. Thus, optimal guidance methods are developed for a satellite operating nearby both non-maneuvering and maneuvering RSOs, while subject to various inspection constraints. Three finite-thrust control types are investigated, as opposed to approximating burns as impulsive, in order increase the accuracy of the generated guidance, which is particularly important for low-thrust systems.

The first control type investigated is a low-thrust, body-fixed engine subject to maximum slew rates, e.g. an electric engine, where the control variables are the throttle level and the three-dimensional direction of the thrust, and mass loss is accounted for. With this control type, pseudospectral methods are used to formulate and solve minimum-time and minimum-fuel multi-phase optimal control problems with respect to a non-maneuvering RSO. The problem combines both a formation establishment and reconfiguration maneuver into one problem, subject to a keep-out cone, where time and fuel can be saved by knowing a priori the desired reconfiguration trajectory and including it in the multi-phase optimal control problem.

The second control type investigated is a satellite with multiple on/off thrusters, capable of thrusting in any direction with the appropriate combination of thrusters. With this control type, analytic expressions are developed for a burn-burn, a burn-coast-burn, and a coast-burn-coast-burn sequence, meaning that the relative states can be propagated analytically through the sequence given the three-dimensional direction and duration of each burn. These are then used within optimization solvers

to reach desired trajectories about a non-maneuvering RSO, where both the natural motion circumnavigation (NMC) and teardrop relative trajectories are thoroughly studied. Due to the analytic propagation of the sequences and the low number of optimization variables required, CPU times for metaheuristic solvers are fast. Thus, particle swarm and genetic algorithms are used to quickly produce reliable initial guesses for a nonlinear programming problem solver to further refine. The maneuvers investigated can be subject to multiple constraints, to include: sunlight constraints, field-of-view constraints, and active and passive collision avoidance constraints.

The third control type investigated is a constantly on, steerable thruster (e.g. an electric engine), where the RSO uses this control type as well and optimally maneuvers away from the inspector satellite. The problem is formulated and solved as a differential game — a zero-sum pursuit-evasion game — in order to find worst case scenarios for the players. Multiple games are developed with respect to an elliptical orbit, in case the maneuvering RSO has departed its circular orbit, to reduce error incurred by any eccentricity. The following games are formulated and solved with metaheuristic methods, where each game corresponds to an inspection goal of the inspector satellite: a) intercept; b) rendezvous; c) obtain Sun vector; d) match energy; e) obtain Sun vector and match energy; and f) match energy and remain close (during the ensuing orbit). Therefore, open-loop strategies can be obtained for these games, where the pursuer wishes to minimize the time to obtain these goals, while the evader wishes to prolong these conditions as long as possible.

Thus, various optimization techniques are used to determine the optimal control for multiple constrained proximity operations inspection maneuvers, where finite thrust is accounted for in all control types considered.

# Acknowledgements

I would first and foremost like to thank my advisor, Dr. Cobb, for his guidance and support throughout my research journey. He always made time to meet with me, thoroughly review my papers, and keep me on track. I'd like to also thank Dr. Fickus for being on my committee, and Dr. Carr and Dr. Hess for their advice and assistance and for being co-authors on several papers. I'd like to also thank Major Curtis, a fellow PhD student, for letting me bounce ideas off of him and discuss topics together, and also the other PhD students who made up our research group. I must also thank my wife and son, for putting up with the long research hours, and understanding the work required for a PhD. Lastly, I'd be amiss if I didn't thank our loyal mini Australian Shepherd for keeping watch while I worked.

Eric R. Prince

*May the Force be with you.*

# Table of Contents

Page

Page

# List of Figures

# List of Tables

# List of Abbreviations

| Abbreviation | | Page |
|---|---|---|
| GEO | geosynchronous orbit | 1 |
| SSA | space situational awareness | 1 |
| RSO | resident space object | 1 |
| GSSAP | Geosynchronous Space Situational Awareness Program | 1 |
| AFRL | Air Force Research Laboratory | 2 |
| ANGELS | Automated Navigation and Guidance Experiment for Local Space | 2 |
| NMC | natural motion circumnavigation | 2 |
| NLP | nonlinear programming problems | 6 |
| GA | genetic algorithm | 7 |
| PSO | particle swarm optimization | 7 |
| LVLH | Local-Vertical, Local-Horizontal | 13 |
| NERMs | general nonlinear equations of relative motion | 14 |
| CNERMs | circular-NERMs | 14 |
| TH | Tschauner-Hempel | 15 |
| YA | Yamanaka and Ankersen | 16 |
| STM | state transition matrix | 16 |
| HCW | Hill-Clohessy-Wiltshire | 20 |
| LROEs | Lovell's relative orbital elements | 22 |
| GCOs | general circular orbits | 25 |
| PCOs | projected circular orbits | 25 |
| TPBVP | two-point boundary value problem | 31 |

| Abbreviation | | Page |
|---|---|---|
| IHM | Indirect Heuristic Method | 31 |
| LPM | Legendre Pseudospectral Method | 32 |
| GPM | Gauss Pseudospectral Method | 32 |
| RPM | Radau Pseudospectral Method | 32 |
| GPOPS-II | General Purpose Optimal Control Software II | 32 |
| SNOPT | Sparse Nonlinear Optimizer | 32 |
| IPOPT | Interior Point Optimizer | 32 |
| SOCS | Sparse Optimal Control Software | 39 |
| NMTs | natural motion trajectories | 46 |
| SDCNLP | semi-direct collocation with nonlinear programming | 57 |
| SDCP | semi-direct control parameterization | 60 |
| MOGA | multi-objective genetic algorithm | 60 |
| ECI | Earth-Centered-Inertial | 112 |
| UTC | Universal Time Coordinated | 112 |
| JD | Julian Date | 112 |
| AUs | astronomical units | 112 |

OPTIMAL FINITE THRUST GUIDANCE METHODS FOR CONSTRAINED SATELLITE PROXIMITY OPERATIONS INSPECTION MANEUVERS

# I. Introduction

## 1.1 Motivation

The geosynchronous orbit (GEO) is prime real estate due to its orbital period matching the Earth's rotational period. Many nations including the United States have critical assets in GEO and are interested in keeping these satellites safe and functioning properly. Thus, GEO space situational awareness (SSA) has become of utmost importance, and it is United States policy to develop SSA information which can be used to detect, identify and attribute actions in space that are contrary to responsible use and the long-term sustainability of the space environment [2]. The United States Department of Defense defines SSA as the requisite current and predictive knowledge of the space environment with one of its key objectives to ensure space operations and spaceflight safety [3]. SSA techniques can be split into two major categories: ground-based and space-based. One of the main advantages of space-based SSA apart from being physically closer to the target(s) is that a space-based asset can obtain global and wide-area coverage over denied areas where little or no data can be obtained from ground and airborne sensors [3]. There are approximately 917* resident space object (RSO)s in the GEO belt [4], and thus one inspector satellite in GEO may have great capability to enhance the knowledge of multiple GEO targets in one mission lifetime [5]. Air Force missions to increase space-based SSA capabilities include the Geosynchronous Space Situational Awareness Program (GSSAP), which has satellites

*as of 22 May 2018

placed in the near-GEO regime supporting U.S. space surveillance operations and allowing more accurate tracking and characterization of man made orbiting objects. These satellites have the advantage of no Earth weather or atmospheric-distortion interruption and have the capability to perform rendezvous and proximity operations [6]. Another mission advancing SSA technology for the Air Force Research Laboratory (AFRL) is the Automated Navigation and Guidance Experiment for Local Space (ANGELS). This mission operates above GEO and is examining techniques to provide a clearer picture of the GEO environment and performs safe, automated operations around its upper stage launch vehicle, testing proximity operation algorithms [7]. The guidance and control for these and future satellites is of course desired to be as optimal as possible, in order to improve the capabilities and extend the mission life of the satellites.

This research thus focuses on producing optimal guidance for an inspector satellite in GEO using various optimization techniques where the term guidance refers to the trajectories the inspector satellite should follow and the control required to generate those trajectories. The target trajectories investigated include the injection into and maintenance of relative motion proximity operations trajectories such as natural motion circumnavigation (NMC) and teardrop* trajectories. The control types investigated are all of finite nature as opposed to impulsive, which is particularly relevant for low-thrust engines in order to produce more accurate guidance. Various inspection constraints are also taken into account, such as keep-out zones and lighting constraints, to generate more applicable results. To solve these complex problems, modern optimization techniques are investigated and applied, such as pseudospectral methods and metaheuristic methods, which are applied to problems formulated both directly and indirectly. Differential game techniques are also employed to account for

*teardrop, or pogo, trajectories refer to those which drift relative to the RSO and form the shape of a teardrop

an RSO which may attempt to optimally delay the goal of the inspector satellite.

## 1.2 Research Questions, Tasks, and Scope

This section describes the overarching research hypothesis and associated research questions. The specific tasks to be completed in order to answer the questions and the research scope are also outlined.

### 1.2.1 Research Questions.

*Hypothesis: By applying modern optimization techniques* to proximity operation maneuvers of an inspector satellite operating in geosynchronous orbit, highly constrained and nonlinear problems can be formulated and solved, realizing time and/or fuel savings and providing mission planners with multiple tools to obtain solutions.*

Research questions addressing this hypothesis include:

1. Problem A: How can an optimal control problem be formulated and solved for a satellite with one, finite-thrust, body-fixed engine and maximum slew rates to start from an arbitrary state nearby and inject itself into an NMC about a non-maneuvering target, and transfer to an orthogonal one in order to reach viewing angles from all eight octants surrounding the target, while adhering to additional inspection constraints?

2. Problem B: How can an optimal control problem be formulated and solved for a satellite with multiple on/off thrusters and capable of reorienting its thrust vector instantaneously, to start from an arbitrary state nearby and inject itself into and maintain a relative teardrop trajectory (as well as into an NMC) about a non-maneuvering target, while adhering to additional inspection constraints?

*such as pseudospectral methods and metaheuristic methods

3. Problem C: How can a differential game be formulated and solved for a satellite and an uncooperative, maneuvering RSO, both with a constant, steerable thruster, to find the zero-sum, optimal, open-loop strategies for the pursuer to achieve its inspection goal as soon as possible from an arbitrary state nearby, while the evader prolongs it as long as possible, for various types of inspection goals?

4. Can metaheuristic optimization algorithms be applied reliably to produce better initial guesses and/or complete, comparable solutions for the proximity operations guidance problems investigated in this research and provide planners another tool to assist in obtaining optimal guidance?

For ease of reference throughout the document, question one is denoted as Problem A, question two is denoted as Problem B, and question three is denoted as Problem C. These problems are distinguished mainly by the control type used. Question four does not represent a specific problem, but has been posed to emphasize the desire to apply metaheuristic methods to help solve the other questions.

### 1.2.2 Research Tasks.

To answer the research questions pertaining to the hypothesis, the following tasks will be performed:

1. Use pseudospectral methods to formulate and solve Problem A. Generate methods to find both minimum-time and minimum-fuel solutions subject to an exclusion cone attached to the RSO.

2. Use various optimization techniques, to include metaheuristic optimization algorithms, to formulate and solve Problem B for varying levels of fidelity. Generate methods to find both minimum-time and minimum-fuel solutions subject

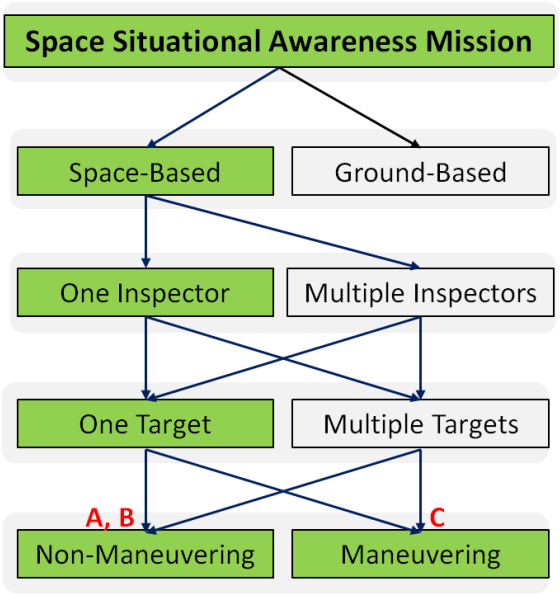
to various inspection constraints, such as lighting and collision constraints.

3. Use metaheuristic optimization algorithms to solve indirect formulations of Problem C, for various inspection goals.

### 1.2.3 Research Scope.

This research addresses the SSA mission of a single space-based inspector satellite inspecting one RSO. Although there exist many ways to inspect a GEO RSO with a space-based asset, this research only considers a few particular inspection missions. Specifically, the first type of inspection mission considered is an initial inspection of an RSO, where the inspector is injected into different NMCs and constrained to avoid an exclusion cone emitting from the satellite pointing towards nadir to avoid interfering with its operations. This type of mission would be one where minimal information exists regarding the RSO with the goal of obtaining views from all eight octants surrounding the target. This inspection mission is investigated in Problem A. Another type of inspection mission investigated in this research is one where the inspector is required to hover in a specified region with respect to the target in order to collect information from one specific set of angles. It may also be allowed to drift by the target, but required to spend a large percentage of its time near one set of angles with respect to the target. This type of collection or inspection mission can be accomplished with a relative teardrop trajectory and is thoroughly investigated in Problem B. Problems A and B both take place with respect to a non-maneuvering RSO. Problem C differs not only in the control type used, but also because the type of inspection mission examined in Problem C is one where the RSO of interest is expected to maneuver optimally to delay the objective of the inspector satellite. Although other types of inspection missions exist (and become more specific depending on exact mission requirements), this research considers the types of missions and rel-

ative trajectories explained above and summarized in Figure 1, where the missions and trajectories highlighted in green are the ones addressed in this research and the research problems have been placed where they apply. Thus Problem A investigates maneuvers into NMCs about a non-maneuvering RSO, Problem B investigates maneuvers into teardrops and NMCs about a non-maneuvering RSO, and Problem C investigates maneuvers into the same orbit as an optimally maneuvering RSO.




(a) Space-Based SSA Mission Types

| Proximity Inspection Trajectories | |
|---|---|
| Fuel Efficient: | Fuel Demanding: |
| A, B NMC (energy matched) | B Teardrop (maintained) (different orbit) |
| B Teardrop (flyby) (different orbit) | Hover (different orbit) |
| C Leader-Follower (energy matched) | Fast Motion Circumnavigation |

(b) Proximity Inspection Trajectories

**Figure 1. Space-Based SSA Mission and Inspection Trajectory Types**

The optimal control problems considered in this research are generally too complex to be solved analytically and must be calculated numerically. Many numerical methods exist to solve optimal control problems, such as direct and indirect methods, as well as metaheuristic methods and hybrid combinations of these methods. Direct methods typically involve the discretization, or parameterization, or approximation of the control and/or states, and result in a static or parameter optimization problem. The resulting optimization problems are typically nonlinear programming problems (NLP)s, and may be solved with existing NLP solvers. Indirect methods apply calcu-

lus of variations and result in equations which form a boundary value problem, which is typically solved via a numerical method. Direct methods are used in Problems A and B, and indirect methods are used in Problem C. Typically, due to its complexity, an optimal control problem is discretized via some method (e.g. pseudospectral methods), which in turn generates an NLP. This technique is used in Problem A. However, if an optimal control problem can formulated in such a way that it is naturally a static optimization problem, then the resulting problem may be solved immediately with an NLP solver, as opposed to using pseudospectral methods to first transcribe the optimal control problem to an NLP beforehand. For such a problem, metaheuristic methods such as a genetic algorithm (GA) or particle swarm optimization (PSO) may be used to solve the static optimization problem and serve as an initial guess. This approach is investigated in Problem B. With indirect methods, the resulting boundary value problem may be solved with a optimization solver. This approach is used for Problem C, where metaheuristic methods are used to solve the boundary value problem. Other techniques exist to formulate and solve optimal control problems, but the ones discussed in this paragraph are explored in this research. They will be described further in Chapter II and it will be shown how they apply to each problem in the following chapters.

This research addresses methods to generate optimal guidance, or optimal open-loop control, which process is also known as path planning or trajectory generation. Thus, the resulting algorithms could be used within a mission planning tool to generate optimal guidance given fuel and timing requirements for a specific mission. Obviously, feedback controllers or closed-loop controllers will be necessary on orbit to correct for model uncertainties, sensor noise, and perturbations. This research, however, does not address such controllers and instead has the purpose of increasing the fidelity of and providing new algorithms for the path planning of multiple

phase, highly constrained, and nonlinear problems. This research is thus composed of algorithmic development and numerical simulations and does not address the implementation of the resulting algorithms on hardware.

### 1.3 Assumptions and Limitations

One of the main reasons to use modern optimization tools is because they have the potential to solve nonlinear, non-convex, and highly constrained optimization problems and generally don't require as many assumptions as other methods, thus producing better results. For example, the research conducted here accounts for finite-duration burns, as opposed to impulsive burns. Mass loss is also accounted for in most cases, and not assumed constant. However, certain assumptions do apply when, for example, linearized relative equations of motion are used as the dynamical model for relative spacecraft motion. In these cases, the RSO, or chief satellite, may need to be in a circular orbit (which applies for the GEO RSOs of interest), the motion of the deputy or inspector with respect to the target must remain in the neighborhood of the target, and the motion cannot be propagated too far forward in time. Otherwise, the linearized equations of motion may no longer be valid.

Also, as discussed, the trajectories generated by these optimization techniques are generally the open-loop, initial trajectories and will need to be accompanied by closed-loop, feedback controllers to keep the satellite on the planned trajectory. Thus, the results generated are as good as the model, which may be fairly detailed, and as good as the perturbations and disturbances which are incorporated into the overall model. The knowledge of where the spacecraft is and will be are also only as good as its sensors, navigation system, and propagation algorithms, which directly tie into the practicality of the optimal guidance generated. Of course, re-planning of the path will be necessary at times. The faster the computation times are, and the higher the trust

is in obtaining stable and convergent results, the more these modern optimization algorithms may be used as part of real-time feedback, or inner-loop, controller.

It must also be understood that the optimization methods used may not result in finding the global minimum. This is the case for non-convex problems, which this research considers. Metaheuristic methods may help with this problem, as one of their benefits is that, (depending on the problem and formulation), they can explore a larger percentage of the search space and may be able to more easily avoid converging to a local minimum. Also, the algorithms may not converge at all or may converge to a clearly incorrect answer. Thus, the results generated by these algorithms may need to be checked and verified by a human in-the-loop or by a reliable verification algorithm.

## 1.4 Research Methodology

This research is composed of two main overarching fields, each composed of many subfields: astrodynamics and optimal control or optimization. Obviously, these are vast fields and only certain subfields will be examined. Regarding astrodynamics, research has been conducted to determine the best models and parameters to use for relative satellite motion and their implementation into optimization algorithms. Using these models, equations are developed to describe the target trajectory in terms of terminal constraints where possible. That is, terminal constraints are developed which when met, result in the desired unforced trajectory with respect to the RSO. These are developed for both NMC and teardrop trajectories and are a critical part of the optimization problem formulation.

Regarding optimal control or optimization, modern optimization tools will be used to solve the complex optimization problems which this research is designed to answer. Specifically, the latest GAs and PSOs will be used to produce initial guesses for NLP

solvers and solve boundary value problems, and pseudospectral methods will be used to solve multiple phase optimal control problems. Chapters III–V will outline how each of these optimization tools are applied to each problem.

The general research methodology for each Problems A, B, and C, is outlined below.

1. Problem A: Slew-Rate-Limited Guidance

   Pseudospectral methods will be used to account for the unique control type of the inspector satellite, the multi-phase aspect of the problem, and the exclusion cone. The NMC establishment and reconfiguration will be solved as separate problems and then combined into one optimal control problem.

2. Problem B: On/Off Thruster Guidance

   Due to the control type used, the problem will be formulated in a way that it is naturally a parameter optimization problem, composed of relatively few optimization variables. This will enable the use of metaheuristic methods to produce initial guesses for an NLP solver to refine. The solution from this mid-fidelity model may then be used as an initial guess for a pseudospectral method if higher-fidelity solutions are desired.

3. Problem C: Constant, Steerable Thruster Differential Games

   Differential game techniques, specifically the indirect heuristic method which makes use of metaheuristic optimization algorithms, will be used to solve boundary value problems resulting from applying the necessary conditions for a differential game solution. Multiple game conditions will be examined, to include maneuvering into a leader-follower formation with respect to an optimally evading RSO. For these games, it will be assumed that both the inspector satellite and the RSO employ a thruster which is always on at the maximum acceleration

magnitude, e.g. an electric engine, where the direction of the acceleration vector may vary. Also, equations of motion will be used which allow the maneuvering RSO to be in a non-circular orbit.

Chapters III, IV, and V will thus cover each problem and present how each optimal control or optimization problem is formulated, and how the mentioned optimization tools can be used to obtain a solution. Results obtained from different optimization techniques will be compared and pros and cons will be assessed regarding each solution technique. The lessons learned and the results will be presented, which will answer the research questions and provide evidence to support the research hypothesis.

## 1.5 Expected Contributions

This research is expected to answer the research questions outlined in Section 1.2.1 by following the methodology to complete the research tasks. The expected contributions will in general increase the capabilities of optimal guidance algorithms for an inspector satellite operating near GEO, and show that time and/or fuel-optimal guidance can be obtained for complex maneuvers by applying modern optimization techniques to problems which account for more realistic control types and constraints. Thus the expected contributions are optimization problem formulations and algorithms which solve Problems A, B, and C, where the algorithms are reliable and computationally fast. The specific contributions from each problem will be shown in Chapter VI.

The efficacy of the results will be measured via several techniques. For multi-phase problems, the multi-phase solution will be compared against the solution obtained when the phases are optimized separately. The effectiveness of the results will also be measured by comparing them against results obtained with added assumptions. For example a finite-burn solution will be compared against an impulsive burn solution.

Also, some methods will be verified by running multiple simulations, to provide empirical evidence that an algorithm produces good results or that a solution has been found. These techniques will be used throughout Chapters III–V to help validate the contributions.

## 1.6 Document Preview

This document is divided into six chapters. This first chapter serves as an introduction. Chapter II contains background information and a literature review detailing recent work performed in the areas this research will focus on, with the purpose of exposing limits in those specific fields which this research is intended to extend. Chapters III–V discuss the methodology and results for Problems A, B, and C respectively, and are composed of the following sections where they apply: problem overview, problem environment, guidance technique, target trajectories, constraints, initial guess methods, optimization problem formulations, simulations and results, and validation and verification of the developed techniques. Chapter VI summarizes the research and outlines the research contributions as well as potential future work.

# II. Background

## 2.1 Overview

This chapter is intended to provide the background and previous studies upon which this work builds in order to answer the proposed research questions. Relative satellite equations of motion, their solutions, and parameterizations are reviewed, with special attention on previous work done to more easily describe and characterize certain types of relative motion. The optimal control problem is formally introduced, along with several techniques and tools used to solve them. Recent research is then presented where these optimization tools have been used to solve problems related to those addressed in this research, in order to show how this work builds upon previous work and how this work extends these specific fields of research. Differential games and pursuit-evasion optimization techniques are also introduced, as well as recent applications of the theory to problems related to this research. The research gaps which this research is intended to fill are then summarized.

## 2.2 Relative Satellite Motion

The equations describing the motion of an inspector satellite (or deputy) relative to an RSO (or chief) have been derived under a variety of assumptions. Typically, the equations are expressed in a non-inertial, rotating, coordinate frame fixed to the RSO with the RSO at its center, where the $\hat{x}$ axis points in the direction from the center of the Earth to the RSO, the $\hat{z}$ axis points in the same direction as the specific angular momentum vector, and the $\hat{y}$ axis completes the right-handed coordinate system. This frame is commonly called the Local-Vertical, Local-Horizontal (LVLH) frame. Using this coordinate system and assuming Keplerian, two-body motion, the

general nonlinear equations of relative motion (NERMs) can be expressed as [8]:

$$\ddot{x} - 2\dot{f}\dot{y} - \ddot{f}y - \dot{f}^2 x + \frac{\mu\left(r+x\right)}{\left[\left(r+x\right)^2 + y^2 + z^2\right]^{\frac{3}{2}}} - \frac{\mu}{r^2} = a_x \tag{1}$$

$$\ddot{y} + 2\dot{f}\dot{x} + \ddot{f}x - \dot{f}^2 y + \frac{\mu y}{\left[\left(r+x\right)^2 + y^2 + z^2\right]^{\frac{3}{2}}} = a_y \tag{2}$$

$$\ddot{z} + \frac{\mu z}{\left[\left(r+x\right)^2 + y^2 + z^2\right]^{\frac{3}{2}}} = a_z, \tag{3}$$

where $f$ is the true anomaly of the RSO, $\mu = 398{,}600.5$ km$^3$/s$^2$ is the Earth's gravitational parameter, $r$ is the distance from the center of the Earth to the RSO, and $a_x$, $a_y$, and $a_z$ are the acceleration terms resulting from any non-Keplerian forces acting on the inspector satellite. If the RSO is in a circular orbit or near circular, then the equations reduce to the circular-NERMs (CNERMs):

$$\ddot{x} - 2\omega\dot{y} - \omega^2 x + \frac{\mu\left(a+x\right)}{\left[\left(a+x\right)^2 + y^2 + z^2\right]^{\frac{3}{2}}} - \frac{\mu}{a^2} = a_x \tag{4}$$

$$\ddot{y} + 2\omega\dot{x} - \omega^2 y + \frac{\mu y}{\left[\left(a+x\right)^2 + y^2 + z^2\right]^{\frac{3}{2}}} = a_y \tag{5}$$

$$\ddot{z} + \frac{\mu z}{\left[\left(a+x\right)^2 + y^2 + z^2\right]^{\frac{3}{2}}} = a_z, \tag{6}$$

where $\dot{f} = \omega$ is constant and is the mean motion of the RSO, and $r = a$, the semi-major axis of the RSO's orbit.

### 2.2.1 Tschauner-Hempel Equations of Motion.

If the distance between the deputy and chief is much smaller than the distance between the center of the Earth and the chief, then the NERMs, Equations 1–3,

reduce to [8]:

$$\ddot{x} - 2\dot{f}\dot{y} - \ddot{f}y - \dot{f}^2 x - 2\mu x \left(\frac{\dot{f}}{h}\right)^{\frac{3}{2}} = a_x \tag{7}$$

$$\ddot{y} + 2\dot{f}\dot{x} + \ddot{f}x - \dot{f}^2 y + \mu y \left(\frac{\dot{f}}{h}\right)^{\frac{3}{2}} = a_y \tag{8}$$

$$\ddot{z} + \mu z \left(\frac{\dot{f}}{h}\right)^{\frac{3}{2}} = a_z, \tag{9}$$

where $h$ is the constant specific angular momentum of the RSO. If the independent variable is changed from time to the true anomaly, $f$, of the RSO and the variables are transformed according to

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix} = (1 + e\cos f) \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{10}$$

where $e$ is the eccentricity of the RSO's orbit, then a simplified set of equations can be derived, known as the Tschauner-Hempel (TH) equations of motion [9]:

$$\tilde{x}'' = \frac{3}{k}\tilde{x} + 2\tilde{y}' \tag{11}$$

$$\tilde{y}'' = -2\tilde{x}' \tag{12}$$

$$\tilde{z}'' = -\tilde{z}', \tag{13}$$

where $a_x$, $a_y$, and $a_z$ have been set equal to zero, $\frac{\mathrm{d}()}{\mathrm{d}f} = ()'$, and $k = 1 + e\cos f$.

Equations 11–13 form a linear, 'time'-varying (i.e. true anomaly-varying) set of differential equations, since $k$ is a function of the true anomaly of the RSO, and can

be written as

$$\tilde{X}' = A(f)\tilde{X}, \tag{14}$$

where $\tilde{X} = [\tilde{x}, \tilde{y}, \tilde{z}, \tilde{x}', \tilde{y}', \tilde{z}']$. Yamanaka and Ankersen (YA) found a simple and singularity-free solution to this set of differential equations, where the solution is

$$\tilde{X}(f) = \Theta(f, f_0)\tilde{X}(f_0), \tag{15}$$

where the state transition matrix (STM) is [10]

$$\Theta(f, f_0) = \begin{bmatrix} s & 2-3esI & 0 & c & 0 & 0 \\ c\left(1+\frac{1}{k}\right) & -3k^2I & 0 & -s\left(1+\frac{1}{k}\right) & 1 & 0 \\ 0 & 0 & \cos f & 0 & 0 & \sin f \\ s' & -3e\left(s'I+\frac{s}{k^2}\right) & 0 & c' & 0 & 0 \\ -2s & -3(1-2esI) & 0 & e-2c & 0 & 0 \\ 0 & 0 & -\sin f & 0 & 0 & \cos f \end{bmatrix}_{f=f} \times \frac{1}{\eta^2} \begin{bmatrix} -3s\frac{k+e^2}{k^2} & 0 & 0 & c-2e & -s\frac{k+1}{k} & 0 \\ 3k-\eta^2 & 0 & 0 & es & k^2 & 0 \\ 0 & 0 & \eta^2\cos f & 0 & 0 & -\eta^2\sin f \\ -3\left(e+\frac{c}{k}\right) & 0 & 0 & -s & -\left(c\frac{k+1}{k}+e\right) & 0 \\ -3es\frac{k+1}{k^2} & \eta^2 & 0 & -2+ec & -es\frac{k+1}{k} & 0 \\ 0 & 0 & \eta^2\sin f & 0 & 0 & \eta^2\cos f \end{bmatrix}_{f=f_0}, \tag{16}$$

where for clarity it has been emphasized that the first matrix in Equation 16 is evaluated at $f$ and the second matrix is evaluated at $f_0$, i.e., every $f$ explicitly appearing

and any functions of $f$ should be $f_0$ and functions of $f_0$ respectively. The terms inside Equation 16 are defined as follows:

$$k = 1 + e\cos f \tag{17}$$

$$s = k\sin f \tag{18}$$

$$c = k\cos f \tag{19}$$

$$I = \frac{h}{p^2}(t - t_0) \tag{20}$$

$$\eta = \sqrt{1 - e^2}, \tag{21}$$

where $p$ is the semi-latus rectum of the RSO,

$$p = a\eta^2. \tag{22}$$

The TH equations of motion and the YA STM can be used to target trajectories with impulsive $\Delta V$s. Most of the relative motion trajectories of interest are bounded in the relative frame and remain close to the origin of the relative frame, meaning that the inspector satellite stays close to and has periodic motion with respect to the RSO. In order to bound the relative motion, the energy of the inspector satellite must match the energy of the RSO. This is called the energy matching condition and has been shown for the TH equations of motion to be [11]:

$$(2 + 3e\cos f_f + e^2)\tilde{x}_f + e\sin f_f(1 + e\cos f_f)\tilde{x}'_f + (1 + e\cos f_f)^2\tilde{y}'_f = 0, \tag{23}$$

where the subscript '$f$' denotes the final value after a given maneuver. If these conditions hold at the end of a maneuver and thus at the beginning of the ensuing natural motion, then the motion of the inspector satellite will be bounded with respect to the RSO. Given this condition, Sengupta parameterized the solution to the TH

equations of motion in order to more clearly describe the geometry of the periodic relative motion:

$$\tilde{x} = \frac{\varrho_1}{p}\sin(f+\alpha_0)(1+e\cos f) \tag{24}$$

$$\tilde{y} = \frac{\varrho_1}{p}\cos(f+\alpha_0)(2+e\cos f) + \frac{\varrho_2}{p} \tag{25}$$

$$\tilde{z} = \frac{\varrho_3}{p}\sin(f+\beta_0), \tag{26}$$

where the relative orbit parameters, $\varrho_1$, $\varrho_2$, $\varrho_3$, $\alpha_0$, and $\beta_0$ are:

$$\varrho_1 = \frac{a}{\eta}(\eta^2\delta e^2 + e^2\delta M_0^2)^{\frac{1}{2}} \tag{27}$$

$$\varrho_2 = p\left(\delta\omega_{\text{œ}} + \delta\Omega\cos i + \frac{1}{\eta^3}\delta M_0\right) \tag{28}$$

$$\varrho_3 = p(\delta i^2 + \delta\Omega^2\sin^2 i)^{\frac{1}{2}} \tag{29}$$

$$\alpha_0 = \tan^{-1}\left(-\frac{\eta}{e}\frac{\delta e}{\delta M_0}\right) \tag{30}$$

$$\beta_0 = \tan^{-1}\left(-\frac{\delta\Omega\sin i}{\delta i}\right) + \omega_{\text{œ}}, \tag{31}$$

where the differential orbital element vector $\delta_{\text{œ}} = [\delta a, \delta e, \delta i, \delta\Omega, \delta\omega_{\text{œ}}, \delta M_0]$ contains the orbital element differences (to the first order) with respect to the RSO of the semi-major axis, eccentricity, inclination ($i$), right-ascension of the ascending node ($\Omega$), argument of perigee ($\omega_{\text{œ}}$), and initial mean anomaly ($M_0$). Relating these differential

orbital elements back to the TH states, Sengupta showed:

$$\delta a = \frac{2a}{\eta^2} c_3 \tag{32}$$

$$\delta M_0 = \frac{\eta^3}{e} c_2 \tag{33}$$

$$\delta e = -\eta^2 c_1 \tag{34}$$

$$\delta i = \sin(\omega_{\text{œ}} + f_f)\tilde{z}_f + \cos(\omega_{\text{œ}} + f_f)\tilde{z}'_f \tag{35}$$

$$\delta\Omega = \frac{-\left[\cos(\omega_{\text{œ}} + f_f)\tilde{z}_f - \sin(\omega_{\text{œ}} + f_f)\tilde{z}'_f\right]}{\sin i} \tag{36}$$

$$\delta\omega_{\text{œ}} = c_4 - \frac{\delta M_0}{\eta^3} - \delta\Omega\cos i, \tag{37}$$

where,

$$\begin{aligned} c_1 = & -\frac{3}{\eta^2}(e + \cos f_f)\tilde{x}_f - \frac{1}{\eta^2}\sin f_f(1 + e\cos f_f)\tilde{x}'_f \\ & -\frac{1}{\eta^2}(2\cos f_f + e + e\cos^2 f_f)\tilde{y}'_f \end{aligned} \tag{38}$$

$$\begin{aligned} c_2 = & -\frac{3}{\eta^2}\frac{\sin f_f(1 + e\cos f_f + e^2)}{1 + e\cos f_f}\tilde{x}_f + \frac{1}{\eta^2}(\cos f_f - 2e + e\cos^2 f_f)\tilde{x}'_f \\ & -\frac{1}{\eta^2}\sin f_f(2 + e\cos f_f)\tilde{y}'_f \end{aligned} \tag{39}$$

$$c_3 = (2 + 3e\cos f_f + e^2)\tilde{x}_f + e\sin f_f(1 + e\cos f_f)\tilde{x}'_f + (1 + e\cos f_f)^2\tilde{y}'_f \tag{40}$$

$$c_4 = -\frac{1}{\eta^2}(2 + e\cos f_f)\left[\frac{3e\sin f_f}{1 + e\cos f_f}\tilde{x}_f + (1 - e\cos f_f)\tilde{x}'_f + e\sin f_f\tilde{y}'_f\right] + \tilde{y}_f, \tag{41}$$

where $c_3$ has already been set equal to zero in Equation 23 by enforcing the energy matching condition. The size of the unforced relative motion can thus be prescribed by choosing $\varrho_1$ and $\varrho_3$, the bias can be set with $\varrho_2$, and the phase angles can be set with $\alpha_0$ and $\beta_0$.

### 2.2.2 Hill-Clohessy-Wiltshire Equations of Motion.

If the distance between the deputy and chief is much smaller than the distance between the center of the Earth and the chief, then the CNERMs, Equations 4–6, can be reduced to the Hill-Clohessy-Wiltshire (HCW) equations of motion [12]:

$$\ddot{x} - 2\omega\dot{y} - 3\omega^2 x = a_x \tag{42}$$

$$\ddot{y} + 2\omega\dot{x} = a_y \tag{43}$$

$$\ddot{z} + \omega^2 z = a_z. \tag{44}$$

Note that for a circular orbit the $\hat{y}$ direction points in the direction of the velocity vector of the RSO, or the in-track direction, while $\hat{x}$ still points in the radial direction and $\hat{z}$ still points in the cross-track direction. A detailed derivation of the solution to the homogeneous HCW equations, i.e. with $a_x$, $a_y$, and $a_z = 0$, can be found in sources such as Vallado [13] and is known to be:

$$x(t) = \left(4x_0 + \frac{2\dot{y}_0}{\omega}\right) - \left(3x_0 + \frac{2\dot{y}}{\omega}\right)\cos\left[\omega(t-t_0)\right] + \frac{\dot{x}_0}{\omega}\sin\left[\omega(t-t_0)\right] \tag{45}$$

$$\begin{aligned} y(t) = y_0 &- \frac{2\dot{x}_0}{\omega} - (6\omega x_0 + 3\dot{y}_0)(t-t_0) + \frac{2\dot{x}_0}{\omega}\cos\left[\omega(t-t_0)\right] \\ &+ \left(6x_0 + \frac{4\dot{y}_0}{\omega}\right)\sin\left[\omega(t-t_0)\right] \end{aligned} \tag{46}$$

$$z(t) = z_0\cos\left[\omega(t-t_0)\right] + \frac{\dot{z}_0}{\omega}\sin\left[\omega(t-t_0)\right] \tag{47}$$

$$\dot{x}(t) = (3\omega x_0 + 2\dot{y}_0)\sin[\omega(t-t_0)] + \dot{x}_0\cos[\omega(t-t_0)] \tag{48}$$

$$\dot{y}(t) = -(6\omega x_0 + 3\dot{y}_0) - 2\dot{x}_0\sin[\omega(t-t_0)] + (6\omega x_0 + 4\dot{y}_0)\cos[\omega(t-t_0)] \tag{49}$$

$$\dot{z}(t) = -\omega z_0\sin[\omega(t-t_0)] + \dot{z}_0\cos[\omega(t-t_0)], \tag{50}$$

where the subscript "0" indicates the initial value at time $t_0$. The solution can also be expressed with the HCW STM,

$$\Theta(t,t_0) = \begin{bmatrix} 4-3\cos(\omega t) & 0 & 0 & \frac{\sin(\omega t)}{\omega} & \frac{-2(\cos(\omega t)-1)}{\omega} & 0 \\ 6\sin(\omega t)-6\omega t & 1 & 0 & \frac{2(\cos(\omega t)-1)}{\omega} & \frac{4\sin(\omega t)-3\omega t}{\omega} & 0 \\ 0 & 0 & \cos(\omega t) & 0 & 0 & \frac{\sin(\omega t)}{\omega} \\ 3\omega\sin(\omega t) & 0 & 0 & \cos(\omega t) & 2\sin(\omega t) & 0 \\ 6\omega\cos(\omega t)-6\omega & 0 & 0 & -2\sin(\omega t) & 4\cos(\omega t)-3 & 0 \\ 0 & 0 & -\omega\sin(\omega t) & 0 & 0 & \cos(\omega t) \end{bmatrix}, \tag{51}$$

where $t_0$ is assumed to be zero.

A guidance method based on impulsive burns and typically called CW Targeting uses the HCW STM, $\Theta$, to determine the two impulsive burns to reach a prescribed relative state. Given a maneuver time, denoted as $t_f$, and the desired position and velocity, $p_t$ and $v_t$ respectively, the magnitude and direction of two impulsive burns can be determined. If the HCW STM is split up into four $3 \times 3$ matrices, $\Theta_{11}$, $\Theta_{12}$, $\Theta_{21}$, and $\Theta_{22}$, the first impulsive burn is calculated by:

$$\Delta V_1 = \Theta_{12}^{-1}(t_f,t_0)\left(p_t - \Theta_{11}(t_f,t_0)p_0\right) - v_0^- = v_0^+ - v_0^-, \tag{52}$$

where

$$v_0^+ = \Theta_{12}^{-1}(t_f,t_0)\left(p_t - \Theta_{11}(t_f,t_0)p_0\right), \tag{53}$$

where the superscripts '$-$' and '$+$' represent the values before and after the instantaneous $\Delta V$. The inverse of $\Theta_{12}(t_f,t_0)$ should exist for all cases except for when the

following equations equal zero,

$$8\cos(t_f\omega) + 3t_f\omega\sin(t_f\omega) - 8 = 0 \tag{54}$$

$$\sin(t_f\omega) = 0. \tag{55}$$

Thus, at and near every $t_f\omega = n2\pi$ (where $n \in \mathcal{Z}_{\geq 0}$), and at or near the other zeros of the first equation (which only occur beyond $2\pi$) will there be singularities. The second impulsive burn is then calculated by:

$$\Delta V_2 = v_t - \left(\Theta_{21}(t_f, t_0)p_0 + \Theta_{22}(t_f, t_0)v_0^+\right). \tag{56}$$

Lovell and Tragesser re-parameterized the solution to the HCW equations in [14] and refined their results in [15]. These parameters, which will be termed Lovell's relative orbital elements (LROEs), help to characterize the relative motion of the deputy with respect to the chief and provide a clear representation of the geometry of relative motion. Specifically, they applied the Harmonic Addition Theorem to Equations 45–47 and used the *atan2* function, producing equations with constant, periodic, and drifting terms which they defined and denoted as the LROEs. These LROEs can more easily describe the geometry of relative formations, and were derived

to be [14]:

$$a_e = 2\sqrt{\left(\frac{\dot{x}}{\omega}\right)^2 + \left(3x + 2\frac{\dot{y}}{\omega}\right)^2} \tag{57}$$

$$x_d = 4x + 2\frac{\dot{y}}{\omega} \tag{58}$$

$$y_d = y - 2\frac{\dot{x}}{\omega} \tag{59}$$

$$\beta = \text{atan2}(\dot{x}, 3\omega x + 2\dot{y}) \tag{60}$$

$$z_{max} = \sqrt{\left(\frac{\dot{z}}{\omega}\right)^2 + z^2} \tag{61}$$

$$\psi = \text{atan2}(\omega z, \dot{z}), \tag{62}$$

where $a_e$ is the semi-major axis of the instantaneous ellipse in the orbital plane of the RSO, $x_d$ and $y_d$ are the radial and in-track displacements of the instantaneous center from the origin of the relative frame, respectively, $\beta$ is the in-plane phasing angle, $z_{max}$ is the maximum cross-track distance, and $\psi$ is the out-of-plane phasing angle. The Cartesian states can then be defined in terms of the LROEs:

$$x = \frac{-a_e}{2}\cos\beta + x_d \tag{63}$$

$$\dot{x} = \frac{a_e}{2}\omega\sin\beta \tag{64}$$

$$y = a_e\sin\beta + y_d \tag{65}$$

$$\dot{y} = a_e\omega\cos\beta - \frac{3}{2}\omega x_d \tag{66}$$

$$z = z_{max}\sin\psi \tag{67}$$

$$\dot{z} = z_{max}\omega\cos\psi, \tag{68}$$

and the LROEs can then be described as functions of time (still in the absence of

forced motion),

$$a_e = a_{e_0} \tag{69}$$

$$x_d = x_{d_0} \tag{70}$$

$$y_d = y_{d_0} - \frac{3}{2}\omega x_{d_0} t = y_{d_0} - \frac{3}{2}\omega x_d t \tag{71}$$

$$\beta = \beta_0 + \omega t \tag{72}$$

$$z_{max} = z_{max_0} \tag{73}$$

$$\psi = \psi_0 + \omega t, \tag{74}$$

where it can be seen that $a_e$ and $x_d$ are constants of the motion. Lovell and Tragesser then introduced the parameter

$$\gamma = \psi - \beta, \tag{75}$$

which is the constant phase difference between the periodic motion in the orbital plane and the periodic motion in the out-of-plane direction. Introducing this parameter makes it so that the only angle that varies with time is $\beta$:

$$x = \frac{-a_e}{2}\cos\beta + x_d \tag{76}$$

$$\dot{x} = \frac{a_e}{2}\omega\sin\beta \tag{77}$$

$$y = a_e \sin\beta + y_d \tag{78}$$

$$\dot{y} = a_e\omega\cos\beta - \frac{3}{2}\omega x_d \tag{79}$$

$$z = z_{max}\sin(\gamma + \beta) \tag{80}$$

$$\dot{z} = z_{max}\omega\cos(\gamma + \beta). \tag{81}$$

This formulation of the LROEs is very useful to clearly describe the size, location,

and orientation of relative motion trajectories.

Sabol, et al. [16] investigated satellite formation designs using the HCW equations, which designs are referenced frequently in the literature. These formation trajectories are bounded, i.e. they do not drift in the relative frame under the linearized HCW assumptions have the same semi-major axis as that of the chief. The constraint which ensures this is easily seen from the HCW equations,

$$\dot{y_f} = -2x_f\omega. \tag{82}$$

This corresponds to the LROE $x_d$ set equal to zero and results in a projected $2 \times 1$ ellipse in the orbital plane, with the length in the along-track direction twice as long as the width in the radial direction. Sabol et al. explain how six initial conditions or constraints are needed to describe the desired relative motion, and with Equation 82 as the first constraint, the five constraints left to describe the desired motion include the offset of the ellipse in the $y$ direction, the size of the ellipse, the initial location in orbital plane, the amplitude of oscillation in the cross-track direction, and the initial location in the out-of-plane motion. They present conditions which produce in-plane formations, in-track formations, circular formations (which in the literature is commonly referred to as general circular orbits (GCOs)), and projected circular orbits (PCOs).

Bounded relative motion is desirable due to the persistent proximity to the chief without requiring fuel. However, there may be cases where certain types of drifting relative motion may be desired due to mission requirements. One useful type of unbounded relative trajectory investigated by Lovell and Tollefson [17] is the 'pogo' or 'teardrop' trajectory where the satellite is in a quasi-hovering pattern relative to the reference satellite and forms the shape of a teardrop. Lovell and Tollefson developed teardrop parameter equations as simple closed-form expressions of their developed

LROEs to more easily describe these drifting relative trajectories. With the teardrop trajectory existing in the orbital plane of the reference satellite, they show that the geometry of a teardrop trajectory is uniquely determined by only two LROEs, $a_e$ and $x_d$. These two LROEs then define the teardrop geometric parameters as described by Hope and Trask [18]. Thus, if the user wishes to define the teardrop trajectory in terms of these geometric parameters, then a suitable pair (with one of them being $T_p$, or the time in the teardrop) is chosen as the independent variables for the teardrop design variables, and the rest of the parameters are determined. They show that for a teardrop trajectory to exist, the following condition must apply,

$$a_e > \frac{3}{2}|x_d|, \tag{83}$$

where if $x_d > 0$ then the teardrop cusp is on top and the teardrop can be placed above the chief where the opposite applies if $x_d < 0$.

Lovell and Brown extended their analysis [1] and introduced additional geometric properties to describe the teardrop trajectory. Thus, combining five of the parameters introduced by Hope and Trask and the three additional ones introduced by Lovell and Brown, the teardrop parameters (with applicable ones shown in Figure 2 taken from [1]) are shown in Table 1.

**Table 1. Teardrop Parameters**

| Variable | Description |
|---|---|
| $R$ | distance from intersection point of teardrop to RSO |
| $\Delta R$ | teardrop height (from intersection point to closest approach) |
| $W$ | maximum teardrop width |
| $T_p$ | period of motion once around teardrop pattern |
| $\Delta V$ | magnitude of impulse required to repeat teardrop |
| $D$ | distance between closest approach to RSO and RSO |
| $E$ | distance from RSO to inspector satellite when in straight-line approach to teardrop |
| $\bar{x}$ | time-averaged distance from RSO to inspector satellite while in teardrop |

They show that $\bar{x}$ is a very useful design parameter, and the total $\Delta V$ required

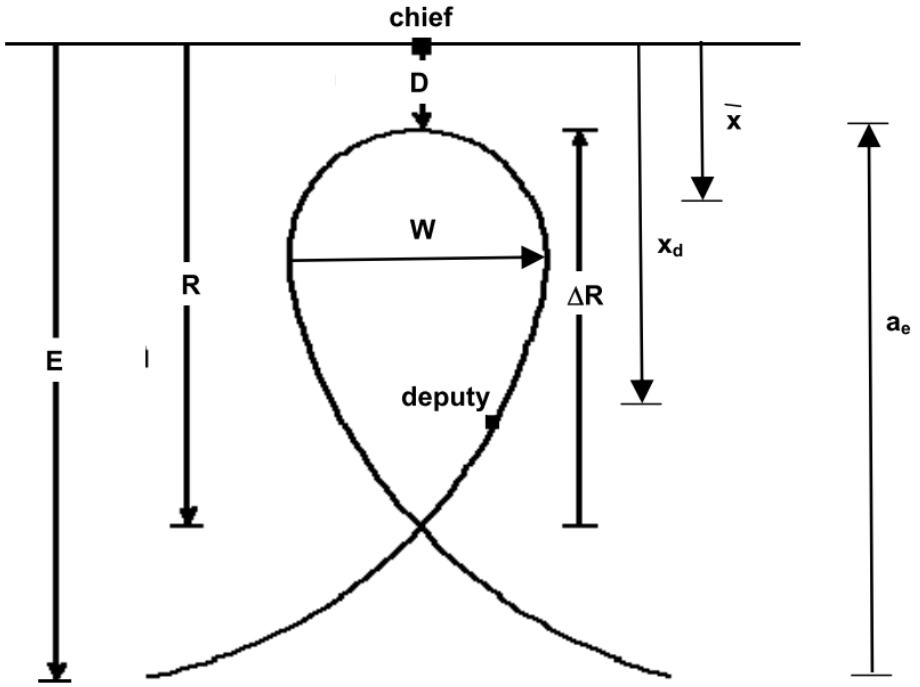



**Figure 2. Applicable Teardrop Parameters (taken from Lovell [1])**

to hover at $\bar{x}$ is equal to the impulsive $\Delta V$ required to maintain a teardrop with the same value of $\bar{x}$, that is,

$$\Delta V_{\Delta t} = 3\omega^2|\bar{x}|\Delta t = \Delta V_{\Delta t,\text{cont}}. \tag{84}$$

To show an example of how to design a teardrop trajectory, if mission planners wish to choose for safety the parameter $D$ and then $T_p$, then the two LROEs must be constrained to be:

$$a_e = \frac{6D\gamma_T}{3\gamma_T - 4\sin\gamma_T} \tag{85}$$

$$x_d = -\frac{4D\sin\gamma_T}{3\gamma_T - 4\sin\gamma_T}, \tag{86}$$

where $\gamma_T$ is a function of $T_p$

$$\gamma_T = \frac{\omega T_p}{2}. \tag{87}$$

Then, the rest of the teardrop parameters are defined:

$$\Delta R = \frac{3D\gamma_T}{3\gamma_T - 4\sin\gamma_T}(1 - \cos\gamma_T) \tag{88}$$

$$R = \frac{3D\gamma_T\cos\gamma_T - 4D\sin\gamma_T}{3\gamma_T - 4\sin\gamma_T} \tag{89}$$

$$W = \frac{12D}{3\gamma_T - 4\sin\gamma_T}\left(\sin\gamma_T\left[\cos^{-1}\left(\frac{\sin\gamma_T}{\gamma_T}\right)\right] - \gamma_T\sin\left[\cos^{-1}\left(\frac{\sin\gamma_T}{\gamma_T}\right)\right]\right) \tag{90}$$

$$\Delta V = \frac{6\omega D\gamma_T\sin\gamma_T}{3\gamma_T - 4\sin\gamma_T} \tag{91}$$

$$E = -\left(\frac{3\gamma_T + 4\sin\gamma_T}{3\gamma_T - 4\sin\gamma_T}\right)D \tag{92}$$

$$\bar{x} = -\frac{D\sin\gamma_T}{3\gamma_T - 4\sin\gamma_T}. \tag{93}$$

The equations for $a_e$ and $x_d$ and the dependent teardrop parameters, given each pair of independent teardrop parameters (with $T_p$ as one of them), can be found in [17] and [1]. Thus the mission planner would choose their two independent teardrop parameters of choice (with $T_p$ as one of them) and the relative teardrop in the orbit plane would be defined.

Lovell and Brown [1] also developed an approximate method so that $T_p$ does not have to be one of the two chosen design variables by making the assumption that $T_p$ is small compared to one orbital period, which makes the intermediate parameter $\gamma_T$ small as well. The truncated Taylor series for sine and cosine can then be used, and variables such as $D$ and $E$ can be chosen as the two independent variables, where $T_p$ becomes of function of them.

## 2.3 Optimal Control and Solution Methods

For a satellite conducting an inspection mission, it is desirable to accomplish maneuvers in an optimal manner which usually corresponds to minimizing the time or fuel to complete the maneuver(s). Thus, the controls of the satellite must be varied such that a specific performance index is minimized, which leads to an optimal control problem. Optimal control theory is a field of active research which has benefited from many contributions over the past several decades. Following Betts's [19], Conway's [20], and Rao's [21] surveys, but adapting the notation to that used in this research, the dynamics of an optimal control problem are typically written as a set of first-order differential equations,

$$\dot{X} = f\left(X(t), u(t), t\right), \tag{94}$$

where $X$ is the state vector, $u$ is the control vector, and $t$ is the independent variable, time. A set of boundary conditions can be described by

$$\psi\left(X(t_0), t_0, X(t_f), t_f\right) = 0, \tag{95}$$

where $\psi$ is the vector of boundary conditions which may have lower and/or upper bounds. Algebraic path constraints may also be present,

$$C_l \leq C\left(X(t), u(t), t\right) \leq C_u, \tag{96}$$

where $C$ is a vector containing the path constraints. There may also be simple bounds on the values of the state and control variables:

$$X_l \leq X(t) \leq X_u \tag{97}$$

$$u_l \leq u(t) \leq u_u. \tag{98}$$

Betts also explains how depending on the chosen performance index, it may be necessary to keep track of integral values,

$$\int_{t_0}^{t_f} \mathcal{L}\left(X(t), u(t), t\right) \mathrm{d}t, \tag{99}$$

where $\mathcal{L}$ is the scalar-valued integrand and is typically called the Lagrangian. Thus, the optimal control problem is to determine the control, $u(t)$, to minimize a chosen cost, which in general has the form

$$J = \phi\left(X(t_0), t_0, X(t_f), t_f\right) + \int_{t_0}^{t_f} \mathcal{L}\left(X(t), u(t), t\right) \mathrm{d}t, \tag{100}$$

subject to the given terminal and path constraints, where $\phi$ represents a scalar function of initial and/or terminal states and time which are to be included in the cost function. Equation 100 represents a general objective function is commonly referred to as the Bolza form.

The referenced surveys [19, 20, 21] describe the optimal control problem and the methodologies used to solve them. Few real-world problems can be solved analytically, in closed form, and thus these surveys focus mostly on numerical methods to solve optimal control problems. Numerical methods can be split into two categories: indirect and direct methods. Indirect methods are those which use the analytic necessary conditions of optimality obtained by applying the calculus of variations to the

optimal control problem described above and typically generate a two-point boundary value problem (TPBVP) which includes the costates as unknowns. This TPVBP may be difficult to solve due to the initial guess required for the non-intuitive costates and their smaller radii of convergence. However, a new solution technique for the classical indirect method will be employed in the research herein, called the Indirect Heuristic Method (IHM), which attempts to solve the resulting TPBVP with a type of metaheuristic optimization algorithm.

Direct methods, on the other hand, transcribe the optimal control problem into a static, or parameter, optimization problem. This is done through the discretization, or parameterization, or approximation of the control and/or states. When the states and/or controls are discretized to points in time, the dynamics and any applicable constraints must be satisfied at each point, creating equality and/or inequality constraints at each discretized point. There are many different types of transcription methods, where one type called pseudospectral methods will be used in this research. Once a transcription method has been applied, the resulting optimization problem is typically an NLP, where an NLP solver is then used to attempt to satisfy the Karush-Kuhn-Tucker conditions [22] in order to find a local minimum.

Metaheuristic methods can be thought of as a separate technique to solve an optimal control problem compared to what indirect and direct methods typically signify, or it can be thought of as a tool to solve indirect or direct formulations. That is, like with IHM, a metaheuristic algorithm can be used to solve the resulting TPBVP by using an indirect method and deriving the first-order necessary conditions by applying the calculus of variations. Or, once the optimal control problem has been transcribed and discretized, i.e. using a direct method, a metaheuristic optimization algorithm can be used instead of an NLP solver to minimize the performance index. Or, perhaps more commonly for metaheuristic methods, the control can be parameterized to

a smaller set of parameters than the resulting time discretization, and the resulting parameter optimization problem can be solved with a metaheuristic optimization algorithm. This usage of metaheuristic optimization algorithms will also be applied to inspection maneuver problems in this research to obtain either an initial guess for an NLP solver or to solve the problem entirely.

### 2.3.1 Pseudospectral Methods.

One type of direct transcription method gaining more traction in the past decade is the pseudospectral method, or the direct orthogonal collocation method. These methods are advantageous in that their global nature leads to faster convergence times with higher accuracy. Different types of pseudospectral methods exist depending on the exact collocation points used, which are typically the roots of various Legendre polynomials. Pseudospectral methods include the Legendre Pseudospectral Method (LPM) [23], the Gauss Pseudospectral Method (GPM) [24], and the Radau Pseudospectral Method (RPM) [25, 26]. This research employs the RPM via the General Purpose Optimal Control Software II (GPOPS-II) [27], a commercial optimal control software package for MATLAB. The RPM is the most recent choice used by the GPOPS-II developers, with developments of Radau-specific covector mapping theorems proving that the resulting NLP optimality conditions are in fact equivalent to the discretized form of the optimality conditions from the originally posed problem, unlike with the LPM [28]. The RPM uses Radau points as the collocation points, which include interior points based on specific Legendre polynomials and one of the endpoints. According to Garg, this ends up being simpler to implement than the GPM [28]. The resulting NLP from the transcription of the optimal control problem to the Radau points is then solved in GPOPS-II with one of two NLP solvers, Sparse Nonlinear Optimizer (SNOPT) [29] or Interior Point Optimizer (IPOPT) [30]. To

provide a brief overview of how GPOPS-II sets up the problem, following Garg in [28], consider $N$ Radau points, ($\tau_1$, $\tau_2$,...,$\tau_N$) where $\tau_1 = -1$ and $\tau_N < 1$, and a new point defined as $\tau_{N+1} = 1$. Note that the problem has been transformed from $[t_0, t_f]$ to $[-1, 1]$ via the affine transformation,

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2}. \tag{101}$$

Then, the Lagrange polynomials of degree $N$ are given by

$$L_i(\tau) = \prod_{j=1, j\neq i}^{N+1} \frac{\tau - \tau_j}{\tau_i - \tau_j}, \qquad (i = 1, ..., N+1), \tag{102}$$

and the state is approximated by a polynomial of at most degree $N$ using the Lagrange polynomials:

$$X(\tau) \approx Y(\tau) = \sum_{i=1}^{N+1} Y(\tau_i) L_i(\tau), \tag{103}$$

where from here on $Y(\tau_i) = Y_i$. An approximation to the derivative of the states is given by differentiating Equation 103 with respect to $\tau$ and setting it equal to the dynamics at the $N$ Radau points:

$$\dot{X}(\tau) \approx \dot{Y}(\tau) = \sum_{i=1}^{N+1} Y_i \dot{L}_i(\tau_k) = \frac{t_f - t_0}{2} f(Y_k, U_k, \tau; t_0, t_f), \qquad (k = 1, ..., N), \tag{104}$$

where $U_k = U(\tau_k)$. The Radau pseudospectral differentiation matrix, $D$, is then defined with entries as $D_{ki} = \dot{L}_i(\tau_k)$ and the $Y_i$ are stacked into a matrix, $Y^{LGR}$, and thus the previous equation can be expressed as:

$$D_k Y^{LGR} = \frac{t_f - t_0}{2} f(Y_k, U_k, \tau; t_0, t_f), \qquad (k = 1, ..., N), \tag{105}$$

where $D_k$ is the $k$th row of the differentiation matrix. Any path constraints are enforced at the $N$ Radau points,

$$\frac{t_f - t_0}{2} C(Y_k, U_k, \tau; t_0, t_f) \leq 0, \qquad (k = 1, ..., N), \tag{106}$$

and the cost is approximated using Radau quadrature,

$$J = \phi(Y(\tau_1), \tau_1, Y(\tau_{N+1}), \tau_{N+1}) + \frac{t_f - t_0}{2} \sum_{k=1}^{N} w_k \mathcal{L}(Y_k, U_k, \tau; t_0, t_f), \tag{107}$$

where $w_k$ is the quadrature weight associated with the $k$th Radau point. Thus, the resulting NLP is to minimize Equation 107, subject to the inequality constraint(s) in Equation 106, and the equality constraints (dynamics) contained in Equation 105 and any applicable initial or terminal constraints:

$$\psi(Y(\tau_1), \tau_1, Y(\tau_{N+1}), \tau_{N+1}) = 0. \tag{108}$$

### 2.3.2 Metaheuristic Methods.

Metaheuristics or evolutionary algorithms may also be used to solve optimal control problems. These methods are not calculus, gradient-based optimization routines but rather algorithms which attempt to mimic processes in nature to find an optimal solution. They typically include some randomness and thus may be termed stochastic methods as well. Popular evolutionary algorithms include the GA, PSO, ant colony optimization, differential evolution, and simulated annealing. Versions of both the PSO and the GA will be used in this research to produce initial guesses or standalone solutions.

The PSO, created initially by Kennedy and Eberhart [31], mimics a flock of birds searching for food. An optimal control problem can be transcribed to a parame-

ter optimization problem and then solved via an evolutionary algorithm such as a PSO. Following Pontani [32], in an unconstrained parameter optimization problem the goal is to find the $n$ unknown parameters contained in $\chi$ to minimize $J$ where each parameter within $\chi$ has to be assigned lower and upper bounds:

$$\chi_l \leq \chi \leq \chi_u. \tag{109}$$

A particle swarm has $N$ particles where each particle contains a candidate solution (a collection of candidate parameters), called the position vector which is denoted in this research by $\chi(i)$ for the $i$th particle. The 'velocity' vector for each particle $w(i)$ changes the free variables (the parameters to be optimized) from generation to generation to try and find the optimal values for each parameter. The velocities must also be bounded corresponding to the parameter bounds, and special treatment must be taken once a component of the position vector has reached a bound. The key equation is the velocity update equation, which dictates how the position vector for each particle changes from iteration to iteration:

$$w_k^{(j+1)}(i) = c_I w_k^{(j)}(i) + c_C[Z_k^{(j)}(i) - \chi_k^{(j)}(i)] + c_S[Y_k^{(j)} - \chi_k^{(j)}(i)], \tag{110}$$

for $k = 1, ..., n$ (the number of free variables), $i = 1, ..., N$ (the number of particles), and the superscript $j = 1, ...$to the number iterations required. $Z_k^{(j)}(i)$ is the best position ever visited by particle $i$ for the parameter $k$ up to the current iteration $j$, and $Y_k^{(j)}$ is the best position ever visited by the entire swarm for the parameter $k$ up to the current iteration $j$. The three coefficient terms in the velocity update equation, Equation 110, from left to right are the inertia, cognitive, and social terms, ($c_I$, $c_C$, and $c_S$ respectively) which are chosen carefully and multiplied by random numbers to introduce an aspect of randomness. The position vector is then updated through

the position update equation:

$$\chi^{(j+1)} = \chi^{(j)} + w^{(j)}, \tag{111}$$

and the iterations continue until a maximum number of iterations, or, the change in the objective function is less than some tolerance over the past chosen amount of iterations, or some other convergence criteria has been reached.

For the constrained parameter optimization problem, which is more typical of optimal control problems which have been parameterized into a static optimization problem, equality constraints are typically appended to the cost function (when using a PSO):

$$\tilde{J} = J + \sum_{r=1}^{m} W_r |h_r(\chi)|, \tag{112}$$

where there are $m$ equality constraints contained in $h(\chi) = 0$, and the weights $W_r$ must be very carefully chosen. For inequality constraints, a simple way of handling them is to set the cost to infinity if any of the inequality constraints are violated. These are simple and general ways to deal with both equality and inequality constraints and are used in the research herein unless other techniques are required.

Following MATLAB's GA documentation page*, the GA works by first creating a random initial population and then creates new populations at each step by doing the following: 1) scoring each potential solution of the population by finding its cost value; 2) scaling the cost values to convert them into a more usable range of values; 3) selecting parents based on the scaled values; 4) choosing lower-cost members as elite children and passing them to the next generation; 5) producing the rest of the children for the next generation by mutation of a single parent or crossover of a pair

*https://www.mathworks.com/help/gads/how-the-genetic-algorithm-works.html, Accessed 15 April 2018

of parents; and finally 6) replacing the previous population with the children to form the next generation. The algorithm then stops when a certain convergence criteria is reached. Thus, the GA mimics natural selection seen in nature in an attempt to find the optimal solution.

### 2.3.3 Indirect Methods Setup.

Optimal control problems may possibly be solved by applying the calculus of variations, obtaining the first-order necessary conditions, and solving the resulting boundary value problem. One method which will be applied in the research herein is the IHM, coined by Pontani [33]. This method starts like all other indirect methods, by obtaining the first-order necessary conditions. Following [21], it is useful to introduce the augmented Hamiltonian, $\mathcal{H}$:

$$\mathcal{H}(X, \lambda, \mu, u, t) = \mathcal{L} + \lambda^T f - \mu^T C, \tag{113}$$

where $\lambda$ is the vector of adjoint variables, or costates, conjugate to the dynamics, $f$, and $\mu$ is the vector of Lagrange multipliers associated with the path constraints, $C$. The function of terminal conditions, $\Phi$ is also constructed,

$$\Phi = \phi + \nu^T \psi, \tag{114}$$

where the terminal constraint vector, $\psi$, is multiplied by their Lagrange multipliers vector, $\nu$.

Continuing with the notation and general optimal control problem presented in Section 2.3, for a single phase optimal control problem with no static parameters, Rao provides the first-order optimality conditions (with notation adapted to that used in

this research) [21]:

$$\dot{X} = \left[\frac{\partial \mathcal{H}}{\partial \lambda}\right]^T, \qquad \dot{\lambda} = -\left[\frac{\partial \mathcal{H}}{\partial X}\right]^T \tag{115}$$

$$u^* = \arg\min_{u \in \mathbb{U}} \mathcal{H} \tag{116}$$

$$\psi(X(t_0), t_0, X(t_f), t_f) = 0 \tag{117}$$

$$\lambda(t_0) = -\frac{\partial \phi}{\partial X(t_0)} + \nu^T \frac{\partial \psi}{\partial X(t_0)}, \qquad \lambda(t_f) = \frac{\partial \phi}{\partial X(t_f)} - \nu^T \frac{\partial \psi}{\partial X(t_f)} \tag{118}$$

$$\mathcal{H}(t_0) = \frac{\partial \phi}{\partial t_0} - \nu^T \frac{\partial \psi}{\partial t_0}, \qquad \mathcal{H}(t_f) = -\frac{\partial \phi}{\partial t_f} + \nu^T \frac{\partial \psi}{\partial t_f} \tag{119}$$

$$\mu_j(t) = 0, \qquad when \qquad C_j(x, u, t) < 0, \qquad j = 1, ..., c \tag{120}$$

$$\mu_j(t) \leq 0, \qquad when \qquad C_j(x, u, t) = 0, \qquad j = 1, ..., c, \tag{121}$$

where $\mathbb{U}$ is the feasible control set and $c$ is the number of path constraints. Equation 115 contains the state and costate equations, Equation 116 is known as Pontryagin's Minimum Principle, Equations 117–118 are the state and costate boundary conditions, Equation 119 is the transversality condition, and Equations 120–121 are the complementary slackness conditions. This results in a TPBVP which is typically solved numerically through methods such as the shooting method, the multiple-shooting method, and collocation methods [21]. However, given certain characteristics regarding the resulting system, the IHM may apply, which will be discussed in detail in Section 2.6.

## 2.4 Direct Method Applications Pertaining to Problem A

Now that the basic definition of an optimal control problem and several solution methodologies have been introduced, this section focuses on how direct optimization techniques (especially pseudospectral methods) have been applied to finding the optimal control for relative motion maneuver problems related to Problem A. Thus,

previous work is reviewed where pseudospectral methods are used to account for complex control types, multi-phase maneuvers, and keep-out zones, since Problem A includes all those aspects. After the related work has been reviewed, it will be shown how solving Problem A will extend the field of research.

To begin, a similar problem was studied by Ranieri where he used a direct method via the Sparse Optimal Control Software (SOCS) [34] to find the minimum fuel to perform a fly-by to inspect the nadir-facing side of a target, but stay outside of a cone pointing towards nadir and avoid collision. However, an NMC was not established and only impulsive control was used [35]. Huntington, et al. showcased the capabilities of the GPM and used it to find the minimum fuel to form a tetrahedral formation for four satellites from a parking orbit, where both one and two allowable maneuvers per spacecraft were examined. They used finite-burn phases along with coast phases, and ensured formation establishment by setting specific terminal constraints, to include: 1) the mesocenter position and velocity of the formation must coincide with the desired reference orbit apogee position and velocity; 2) the average formation side length must lie in a certain range; 3) the tetrahedron shape must adhere to a lower bound on something called the Geometric Factor; and 4) the periodicity of the formation must ensure all four satellites have the same semi-major axis at the final time [36]. They also studied tetrahedral formation reconfiguration, where the fuel was minimized and the effects of $J_2$ were incorporated, and it was ensured that the formation stayed in shape for a certain region of interest [37].

Yunhua Wu, et al. coupled translational and rotational dynamics and used the GPM to find the minimum-fuel solution for separate formation establishment and reconfiguration problems. The spacecraft had one, body-fixed thruster with continuous low thrust and reaction wheels, and path constraints formed a rotational keep-out zone ensuring a star tracker camera avoided pointing at the Sun. However, the ter-

minal conditions for each problem forced the satellite to enter an NMC at a certain location or range, and were not free [38].

Baolin Wu, et al. used the LPM to find the minimum-fuel solution using low-thrust control for formation reconfiguration, where the desired new formation was larger compared to the initial, and the point at which the satellites entered the larger formation was free to be optimized. Path constraints included collision avoidance, and in addition to terminal constraints, there were several constraints afterwards during unforced motion to ensure the formation geometry was met [39]. They also used the LPM to find the minimum energy to establish a formation where continuous low-thrust control authority was used along all three translational axes. Collision avoidance was implemented with path constraints and the final formation was established by setting a terminal distance constraint and then minimizing the distance between that terminal position and the position one unforced period later [40].

Ma, et al. also used the LPM to solve a minimum-time problem to reconfigure a formation in deep space with continuous low thrust while avoiding collision, however mass was assumed constant and terminal constraints defined a specific location to enter the reconfigured formation [41]. Lee and Hwang used GPOPS to find the optimal switching times between a high, pseudo-impulsive thruster and a low continuous thruster which resulted in the minimum fuel required to reconfigure a PCO formation to a larger one. The thrusters each had their own authority along all three axes, and the sequence of high vs. low-thrust phases was defined a priori [42]. Inampudi and Schaub used the LPM to find the optimal control to reconfigure a two-craft Coulomb formation in circular orbits, using both Coulomb force and electric microthrusters when necessary. The problem was solved four different times by minimizing time, acceleration of the separation distance, fuel, and electrical power [43].

Huang, et al. used the GPM to find the optimal trajectories for separate for-

mation establishment and reconfiguration problems, and also a rendezvous problem, where the control was the spacecraft charge and the energy used to interact with the Earth's magnetic field was minimized. In the formation establishment and reconfiguration problems, they included a collision avoidance path constraint, and also terminal constraints which defined the desired NMC of the initial and reconfigured formation trajectories. The tilt angle of the NMC orbits with respect to the orbit plane was chosen a priori, but the point at which they entered those NMCs was free [44, 45].

Zengwen, et al. used the GPM via GPOPS to find the minimum-time solution to reconfigure a two-spacecraft formation using electromagnetic actuation. The two satellites started out in the same orbit and reconfigured to a fixed position with velocities only constrained to produce an NMC [46]. And finally, Li recently used the RPM to find the minimum-fuel solution to reconfigure a spacecraft formation using finite burns, where the maneuver was divided up into burn and coast phases [47].

Given the previous work related to Problem A, this research intends to build upon it by introducing a new problem formulation and solution for an inspector satellite to perform an initial inspection of a target satellite with the goal of visiting all eights octants surrounding the target. Specifically, this research presents a formation and reconfiguration problem linked together into one multi-phase optimization problem, meaning for example that the optimal formation establishment solution may change if it sufficiently improves the reconfiguration solution. Like Ranieri, this problem introduces a keep-out cone as an inspection constraint, but introduces a way to use the keep-out cone in the first phase to define two possible NMCs which the satellite may enter to inspect the nadir-facing side of the target. Like Huang, the terminal constraints at one point in time only (as opposed to checking an unforced period and/or two later) allow the inspector to enter the NMCs at whichever point is optimal,

but this research will also allow the optimizer to determine which of the two initial NMCs is optimal instead of selecting which one a priori. The control used will be one, body-fixed, low-thrust engine with maximum slew rates, which like Wu, couples rotational constraints to the translational trajectory, but different in that two angles will be introduced as additional states controlled by their respective angle rates. The thrust magnitude can also be throttled in this research, which allows the optimizer to determine when to thrust vs. coast, instead of defining the burn and coast sequences a priori. Solutions, as will be seen, typically result in constant-magnitude, finite-burn solutions. This work will also take advantage of recent mesh refinement schemes in GPOPS-II which generate a more accurate solution. Thus, the research methodology in Chapter III (for Problem A) will combine several elements existing in previous work into one problem, as well as introduce new aspects as already discussed. The minimum-time solution will be found first, and then the problem will be solved for increasing fixed final times while minimizing fuel to provide mission planners with optimal time vs. fuel options, i.e. the Pareto front of optimal solutions with varying fixed final times.

## 2.5 Previous Work Pertaining to Problem B

This section presents recent research related to Problem B, focusing primarily on guidance methods for hovering or teardrop trajectories. Thus, it reviews previous work where on/off thruster guidance has been investigated, then examines optimization techniques which have been developed for teardrop-like hovering with respect to a target satellite, and quickly discusses previous work pertaining to lighting and collision constraints. Then, this section summarizes how the algorithms developed for Problem B will extend the current field of research.

Typically, when finite, on/off thrusters are used in an optimal control problem,

numerical methods are used to propagate the spacecraft trajectory through a maneuver sequence given the burn starting times, durations, and directions. However, some work was done by Bevilacqua and Lovell where they developed a semi-analytic guidance technique for on/off thrusters by using the HCW equations of motion in conjunction with the LROEs. Specifically, they developed analytic expressions for the LROEs as functions of the on/off control variables, i.e. the number, starting times, durations, and magnitudes of the firings along each axis of the spacecraft [48]. If enough simplifying assumptions are made, then a reduced set of these control variables can be solved for analytically, given the desired LROEs; but in general, these equations would have to be used in conjunction with an optimization tool to find the control variables to reach a desired state or trajectory. The equations developed by Bevilacqua and Lovell might be able to be used successfully within an optimizer; however, they are quite complex and include many instances of the *atan2* function. Thus, the expressions are not ideal for use in gradient-based optimizers. Also, even though the number of control variables can be reduced, it was desirable to develop similar expressions for the HCW states as functions of the fewest number of control variables as possible. This is desired in order for the expressions to be suitable to use within an optimizer, especially within metaheuristic optimizers. Therefore, new expressions will be developed for Problem B to analytically propagate the HCW states, given the control variables of interest.

Regarding optimization techniques which have been developed for relative hovering or teardrop problems, Irvin, Cobb, and Lovell investigated strategies for a deputy satellite to hover within a fixed volume near a chief satellite in a circular orbit using the HCW equations. They found the optimal control using impulsive thrusts with the goal of minimizing fuel per the total time-of-flight desired to stay within a given volume, referred to as a lobe. They compared the fuel required to continuously thrust

and hover vs. discrete-thrust solutions which stay within the defined volume and found that for certain cases the discrete-thrust solution requires less fuel. Thrusting was assumed to occur on the lobe boundary in the orbital plane and their cost function was

$$J = \frac{\sum_{i=1}^{k} \Delta V_i + \Delta V_f}{\sum_{i=1}^{k} T_{i,i+1} + T_f}, \tag{122}$$

where $k$ is the number of legs (or trajectories between burns), $T$ is the time of flight for each trajectory, and the subscript $f$ represents a possible exit burn. They first optimized in the orbital plane and then used $\Delta V$ in the out-of-plane direction to ensure the satellite does not exit the prescribed lobe height. Several cases were examined to include: defined entry with open exit, and open entry with repeating hover where each was analyzed with different types of constraint volumes. One important conclusion they made is that the teardrop trajectory tends to be the lowest cost solution when a repeating hover condition is desired [49, 50].

Williams and Lightsey also investigated optimal impulsive maneuvers in order to remain in close proximity to a target in a circular reference orbit using the HCW equations. They focused on minimizing fuel and maximizing the time-of-flight between maneuver locations, and also considered out-of-plane trajectories. They developed a keep-in zone, also called a lobe, and only allowed maneuvers to take place on a pre-defined maneuver surface. The performance index was:

$$J = \frac{\Delta V^2}{T}, \tag{123}$$

where $\Delta V$ is the fuel needed for the maneuver and $T$ is the total time of flight. They solved single-leg and multiple-leg cases by using a line search algorithm and found the minimum cost subject to a maximum time of flight constraint which would keep

the satellite in the keep-in region [51].

Zhang, et al. [52] examined relative hovering for eccentric orbits, where they determined fuel-optimal positions for stationary hovering. Deaconu, et al. also investigated eccentric hovering and presented a new method for generating optimal impulsive maneuvers for spacecraft proximity operations where they accounted for linear continuous constraints on the trajectory. They parameterized the trajectory and used that parameterization to create a finite convex description of admissible trajectories and then solved the optimal control problem via semidefinite programming. They used Yamanaka and Ankersen's [10] formulation of the relative equations of motion which allow eccentricity and looked at scenarios where relative motion periodicity is required and then isn't required for various proximity operation missions. For hovering missions, they considered reaching a natural, periodic 'hover' trajectory and also cases where a keep-in region constrains the trajectories and thus non-periodic hovering within that region would apply. From an initial state, they found the minimum $\Delta V$ required to reach a state which results in the desired periodic or non-periodic motion, constrained to a region and also constrained by a maximum amplitude for each impulsive firing [53].

The works presented above dealt primarily with hovering within a defined volume or area with respect to the target and only considered impulsive burns. In contrast, the research herein will use finite-duration burns, which becomes necessary to consider for low-thrust propulsion systems and will specifically focus on a satellite with on/off thrusters. This research will also use the parameters developed by Lovell to define the teardrop geometry, and thus the size of the keep-in region in the orbital plane will be specified, as well as the position of the teardrop region along the in-track direction. The desired bounds of the teardrop motion in the out-of-plane direction may be specified as well. Thus, various optimization methods will be used where

finite, on/off thrusters are accounted for to optimally inject an inspector satellite into and maintain desired teardrop trajectories. The same type of control will also be used to maneuver into NMCs, and various inspection constraints will be enforced for each problem. Like Problem A, multiple minimum-time and minimum-fuel problems will be solved, in order to provide mission planners a Pareto front of possible maneuvers.

The inspection constraints that may be enforced in Problem B include lighting and collision avoidance constraints. Regarding lighting constraints, the maneuver may be constrained to enter a teardrop trajectory (or NMC) such that the Sun is illuminating the RSO with respect to the inspector satellite, and where the Moon and the Earth may not appear in the field of view of the inspector satellite. Previous work where such lighting constraints have been incorporated into an optimization problem is limited. One approach was taken by Franquiz, et al., where path constraints were formulated to keep bright objects outside the field of view during observability maneuvers [54]. The research formulation herein differs in several ways, one of which is that the lighting constraints will be incorporated into the terminal constraints, not as path constraints, such that the teardrop (or NMC) motion after the maneuver generates the desired lighting.

Regarding collision avoidance constraints for Problem B, the maneuver may be constrained to be both actively and passively safe, where passively safe means that the inspector satellite would not enter a defined keep-out zone if one of the burns failed to take place. There have been many efforts in the literature to avoid collisions in the relative motion problem. A good review of these methods is contained in the paper by Frey, et al. [55], where they developed an obstacle avoidance approach based on a graph search applied to a virtual net of periodic natural motion trajectories (NMTs). The approach guarantees safe transitions from one NMT to another, and could be used as a warm-start to an optimizer. In Frey's review of obstacle avoidance efforts,

they cited many previous methods investigated, such as Ranieri's method [35] of using SOCS to ensure path constraints are satisfied at each numerically integrated point, which is similar to the way other direct collocation software like GPOPS-II would implement collision avoidance constraints. Other techniques cited by Frey include using artificial potential functions and converting a non-convex optimal control problem into a convex optimal control problem.

The main focus of the research herein is not collision avoidance, but it will be incorporated into Problem B by developing inequality constraints at discrete points in time using the semi-analytic guidance method which will be developed for on/off thrusters. Also, the collision avoidance constraints will be imposed from any arbitrary state to an NMC or teardrop trajectory, and not from one NMT to another. Using the techniques developed herein, the resulting maneuvers will be safe, including passively safe.

## 2.6 Metaheuristic Methods Applications

According to Conway [20], metaheuristic methods have two main advantages over other methods such as classical approaches using indirect and direct methods: 1) they are relatively simpler to code and implement; and 2) they may be more likely to converge to the global minimum. This section focuses on metaheuristic optimization techniques, with special attention dedicated to the particle swarm technique, which have been applied to space trajectory optimization problems in order to provide background for research question number 4. Thus, this section covers applications in general but pays special attention to previous work related to the problems addressed in this research, and shows how metaheuristic methods can be used to solve direct and indirect problem formulations.

Pontani and Conway [32] showed that a PSO could be successfully applied to

various space trajectory optimization problems, to include determining periodic orbits in the restricted three-body problem, a two-impulse transfer between two circular orbits, finite-thrust orbit transfers, and a finite-thrust Earth-to-Mars problem. For the Earth-to-Mars transfer, they applied the necessary conditions for optimality via the calculus of variations and expressed the control as a function of the costates. The cost function for the PSO then simply became a measure of the constraint violations to be minimized. This approach is termed the IHM and will be discussed in detail later in this section. For constrained optimization problems, they penalized constraint violations by utilizing Equation 112, stated again here for convenience:

$$\tilde{J} = J + \sum_{r=1}^{m} W_r |h_r(\chi)|. \tag{124}$$

When handling inequality constraints, if a particle violates an inequality constraint, that particle's velocity is set to zero such that the inertial term in the velocity update equation has no effect and the next velocity update is only affected by the cognitive and social terms.

For problems formulated via a direct method, they parameterized the optimal control problems by representing the time-dependent control in terms of parameters. For the finite-thrust orbit transfer between two circular orbits, they represented the thrust angle as a third-degree polynomial as a function of time during two thrust arcs separated by a coast arc, where the two thrust arcs were parameterized as:

$$\delta = \zeta_0 + \zeta_1 t + \zeta_2 t^2 + \zeta_3 t^3, \qquad 0 \leq t \leq t_1 \tag{125}$$

$$\delta = \vartheta_0 + \vartheta_1 (t - t_2) + \vartheta_2 (t - t_2)^2 + \vartheta_3 (t - t_2)^3, \qquad t_2 \leq t \leq t_f \tag{126}$$

where $\delta$ is the thrust angle in the orbital plane for each burn and the coefficients $[\zeta_0, \zeta_1, \zeta_2, \zeta_3, \vartheta_0, \vartheta_1, \vartheta_2, \vartheta_3]$ must be found by the PSO in order to minimize fuel re-

quired by the two maneuvers [32].

Pontani, et al. also showed how a PSO could be used to find multiple-burn rendezvous trajectories [56]. They applied a PSO to both impulsive and finite-thrust control problems where for the finite-thrust case the control was assumed to be a linear combination of B-splines where the PSO determines the optimal spline degree and coefficients to minimize fuel and constraint violations.

PSOs have been used to solve other types of space trajectory optimization problems, such as responsive theater maneuvers where a global PSO was used by Showalter and Black to solve single-,double-, and triple-pass responsive theater maneuvers with impulsive control [57]. Showalter and Black then used a PSO solution to a similar problem as a seed for an NLP solver when using continuous-thrust control [58]. They also developed optimal cooperative en-route inspections during geostationary transfer maneuvers using hybrid optimal control where the outer-loop metaheuristic solver optimized categorical variables [59].

Regarding relative motion problems, Huang, Zhuang, et al. solved an energy-optimal spacecraft formation reconfiguration problem in deep space using continuous low thrust where mass was assumed constant. They transcribed the optimal control problem to an NLP by applying the LPM and then used particle swarm optimization to solve the NLP. The initial and final states were specified, and they developed a way such that all particles swarm through the hyperplane defined by the set of feasible solutions. They solved the problem with an NLP solver first without considering collision avoidance, and then used the PSO to find the best trajectory which satisfies collision constraints [60]. They refined their strategy in [61] where they used one swarm to represent one satellite and through communication with other swarms avoid collision during the formation reconfiguration. The control and trajectories for each satellite were optimized separately in order to quickly solve the problem and to be

able to use the algorithm in real time. The algorithm examined Labatto points closest to collision and tested points around them to ensure a collision would not occur. They termed this approach the co-evolutionary particle swarm optimization, and is one way to try to handle path constraints when using metaheuristic methods.

Pontani and Conway also applied their techniques to finite-thrust rendezvous trajectories in the relative frame for minimum-time proximity operations and presented solutions for five distinct problems: 1) rendezvous from NMC to target; 2) rendezvous from same circular orbit to target; 3) transfer from same circular orbit to NMC; 4) transfer from one NMC to another NMC; and 5) three-dimensional rendezvous from NMC to target (where the rest of the cases were planar). They used Hamiltonian methods (later termed IHM) to transform the optimal control problems into parameter optimization problems to be solved by a PSO. Equality and inequality constraints were handled in a similar way as before and mass loss was accounted for. To show how the IHM method works and to clearly outline the problem properties allowing the IHM method to be applied, this particular work is reviewed in detail. For these problems, the maneuvering spacecraft is assumed to use a constant, low thrust during the entire time of flight, and thus the problem is to find the thrust direction which results in the minimum-time solution, where $\phi$ is the out-of-plane angle and $\alpha$ is the in-plane angle constrained to $[-\pi/2, \pi/2]$ and $[-\pi, \pi]$ respectively. The cost to be minimized is simply $J = t_f$ with boundary conditions (problem dependent) of the form $\psi(X_0, X_f, t_f) = 0$. Thus the Hamiltonian, $\mathcal{H}$, and function of terminal conditions, $\Phi$, are:

$$\mathcal{H} = \lambda^T f \tag{127}$$

$$\Phi = t_f + \nu^T \psi. \tag{128}$$

The necessary conditions can then be derived where Pontryagin's Minimum Principle

leads to expressions for the optimal control as functions of the costates. The costate equations are

$$\dot{\lambda} = -\left[\frac{\partial \mathcal{H}}{\partial X}\right]^T = -A^T\lambda, \tag{129}$$

where $A$ for this problem is the constant state (or plant) matrix for the unforced HCW equations. The costate boundary conditions are

$$\lambda_0 = -\left[\frac{\partial \Phi}{\partial X_0}\right]^T \quad , \quad \lambda_f = \left[\frac{\partial \Phi}{\partial X_f}\right]^T, \tag{130}$$

and the Hamiltonian at the final time must adhere to

$$\mathcal{H}_f + \frac{\partial \Phi}{\partial t_f} = 0 \implies \mathcal{H}_f + 1 = 0. \tag{131}$$

However, one property that enables the IHM to work is the ignorability of the transversality condition, Equation 131. In order for the transversality condition to be ignorable, the costate boundary equations must be homogeneous in $\lambda$. This, in conjunction with the homogeneity in $\lambda$ of the costate equations, implies that if an optimization algorithm can find an initial value of $\lambda$ such that $\lambda(t_0) = k_\lambda \lambda^*(t_0)$ where $k_\lambda > 0$, then the same proportionality exists between $\lambda$ and the optimal $\lambda^*$ at any time. The control, as stated previously, is a function of the costates but in addition depends only on the relative magnitudes of the costates and is thus equal to the optimal control. Thus, for these problems, satisfying all of the necessary conditions except Equation 131 results in the optimal control, but not the optimal adjoints, which are all scaled by $k_\lambda$. The PSO optimization parameters are thus

$$\chi = [\lambda_{1_0}, \lambda_{2_0}, \lambda_{3_0}, \lambda_{4_0}, \lambda_{5_0}, \lambda_{6_0}, t_f], \tag{132}$$

where the initial adjoints can be sought in the interval $[-1, 1]$ due to this special property applying to these problems. Then, for all particles, the state and costate equations are integrated forward and the optimal control is calculated as a function of the adjoints over time. The boundary condition violations are then evaluated to determine the fitness of each particle. Pontani and Conway successfully applied this algorithm to solve the multiple rendezvous and NMC transfer problems and also showed how the solutions obtained can be used as an initial guess for local numerical solvers [62].

Pontani and Conway then developed a minimum-fuel, finite-thrust, relative motion algorithm where they developed a switching function which determines the optimal sequence and durations of thrust and coast arcs. Applying the necessary conditions to the problem again transformed the optimal control problem into a parameter optimization problem which is then solved by a PSO, which they officially called the IHM. The control variables optimized were again the thrust direction angles $\phi$ and $\alpha$ in three dimensions and also the thrust magnitude via the optimization of the unknown initial values of the adjoint variables. The specific method developed becomes simpler to solve when the initial and final states are defined and special boundary conditions apply (the initial and final $x$ positions and $y$ velocities are equal) but the method is applicable to more general cases as well. They solved five different cases to include: 1) rendezvous from the same circular orbit to the target; 2) same as problem one but with a shorter allowed time; 3) transfer from the same circular orbit to an NMC; 4) a special four-thrust-arc rendezvous problem; and 5) a three dimensional arbitrary rendezvous (where the other four cases were planar) [33]. To clearly describe how the IHM can be applied to minimum-fuel problems, this problem is now described in more detail. The thrust magnitude $T(t)$ is a time varying thrust and is constrained to $[0, T_{max}]$. A seventh state is then defined as $X_7 = m/m_0$ where $m$ is

the time-varying mass and $m_0$ is the initial mass of the spacecraft. If the variable $\tilde{n}$ is defined as $\tilde{n}(t) = T(t)/m_0$ then the seventh state equation can be written as

$$\dot{X}_7 = -\tilde{n}/c, \tag{133}$$

where $c$ is the constant effective exhaust velocity. Then, the complete state vector is the combination of the original six HCW states and the seventh: $\tilde{X} = [X, X_7]$. The cost function for these minimum-fuel problems is

$$J = \int_{t_i}^{t_f} k\tilde{n}(t)\mathrm{d}t = \int_{t_i}^{t_f} \mathcal{L}\mathrm{d}t, \tag{134}$$

where $k$ is an arbitrary positive constant. It is again convenient to define the Hamiltonian, $\mathcal{H}$, and the function of terminal conditions, $\Phi$,

$$\mathcal{H} = \mathcal{L} + \lambda f + \lambda_7 \left(-\frac{\tilde{n}}{c}\right) \tag{135}$$

$$\Phi = \nu^T \psi. \tag{136}$$

From Pontryagin's Minimum Principle, the optimal thrust direction angles are again functions of the the time-varying costates and again only rely on their relative magnitudes. These expressions are then used to develop an expression for $\tilde{n}^*$ (the optimal $\tilde{n}$), where the term typically called the switching function can be extracted to determine the optimal thrust and coast arcs, where during each thrust arc the maximum thrust is used. Given the initial costates, they can be propagated forward by the costate equations, and thus the PSO just needs to find the initial values of the costates. Their values are in general nonintuitive, but a range needs to be set in order to use a PSO. Pontani reasserts his claim here that the search range for the initial adjoints can be set to $[-1, 1]$ due to 1) the fact that $k$ is arbitrary; 2) the fact that the costate equa-

tions for $\lambda$ are homogeneous; and 3) the fact that the optimal control angles resulting from Pontryagin's Minimum Principle are functions of the adjoint variables where these angles remain unchanged if the adjoints are scaled by a positive constant. Also for these minimum-fuel problems, the optimal thrust sequence depends on a relative magnitude of the terms which compose the switching function and thus allowing $k$ to be determined and not set allows the arbitrary range for $\lambda$. Thus $k$ is introduced as an additional parameter for the PSO to optimize with the range of $0.001 \leq k \leq 10$. By taking advantage of the analytical nature of $\lambda$, and re-writing the switching function for this problem as

$$S = \frac{1}{x_7}\left[\left(kx_7 - \frac{\lambda_7}{c}x_7\right) - \sqrt{\lambda_4^2 + \lambda_5^2 + \lambda_6^2}\right], \tag{137}$$

the coast-to-thrust switching times can be searched for in the time intervals that satisfy

$$\lambda_4\dot{\lambda}_4 + \lambda_5\dot{\lambda}_5 + \lambda_6\dot{\lambda}_6 > 0. \tag{138}$$

Then, the set of free variables to be optimized by the PSO are

$$\chi = [\lambda_{1_0}, \lambda_{2_0}, \lambda_{3_0}, \lambda_{4_0}, \lambda_{5_0}, \lambda_{6_0}, \lambda_{7_0}, k], \tag{139}$$

where the PSO takes the following steps for each particle: 1) use Equation 138 to find numerically the time intervals in which coast-to-thrust switching times can occur; 2) express the control via their functions of costates; 3) along thrust arcs, propagate the adjoints analytically and the state equations numerically; 4) along coast arcs, analytically propagate both the adjoints and states; and 5) evaluate boundary condition violations (for both the states and costates) to determine each particle's fitness.

Pontani also wrote a study on symmetry properties of optimal relative orbit trajectories which apply under unique conditions. Both minimum-time and minimum-fuel path generation problems may satisfy these unique requirements for a symmetry property to apply. If it applies, then given an already determined optimal path, a symmetric optimal path is also defined, providing mission planners another trajectory option with the same cost [63].

The application of metaheuristic methods to space trajectory optimization problems presented in this section can be divided into two groups: those applied to solve direct problem formulations and those applied to solve indirect problem formulations. Regarding direct formulations, one research goal will be to find the best way to parameterize the control for on/off thrusters in Problem B to either obtain a good initial guess for an NLP solver or GPOPS-II or to solve the problem entirely, in order to provide mission planners with an alternate tool for solving these problems. The IHM method will also be applied to formulate and solve Problem C, for which the background is discussed in the next section. Thus, the research herein should showcase the ability of these metaheurisitc optimization algorithms to solve the new problems addressed in this research.

## 2.7 Differential Games and Applications

Differential games were formally introduced by Isaacs [64] and treats two-sided optimization of two uncooperative or competing players as opposed to the one-sided optimization or optimal control problems examined up to this point. As explained by Bryson and Ho [65], the general setup of a differential game may be described by starting with the dynamic system of the two players:

$$\dot{X} = f(X, u, v, t), \qquad X(t_0) = X_0, \tag{140}$$

where $u$ is the control of the minimizing player (the pursuer) and $v$ is the control of the maximizing player (the evader), with the terminal constraints

$$\psi(X(t_f), t_f) = 0, \tag{141}$$

and the performance index

$$J = \phi(X(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(X, u, v, t)\mathrm{d}t, \tag{142}$$

where the problem is to find $u^*$ and $v^*$ such that

$$J(u^*, v) \leq J(u^*, v^*) \leq J(u, v^*). \tag{143}$$

This is a zero-sum differential game, meaning that there is one cost function which the pursuer wishes to minimize and the evader wishes to maximize. For a two-person, zero-sum, differential game, following Bryson and Ho, it is desired to find the game-theoretic saddle point solution by applying the first-order necessary conditions. The necessary conditions can be described by first forming the Hamiltonian,

$$\mathcal{H} = \lambda^T f + \mathcal{L}, \tag{144}$$

where again $\lambda$ is the vector of costates, $f$ is the vector containing the system dynamics, and $\mathcal{L}$ is the Lagrangian. The function $\Phi$ is also formulated,

$$\Phi = \phi + \nu^T \psi, \tag{145}$$

where $\nu$ is the vector of Lagrange multipliers associated with the terminal constraints, and $\phi$ is the function of terminal values in the cost function. Given $\mathcal{H}$ and $\Phi$, the

necessary conditions for stationarity are

$$\dot{\lambda} = -\left[\frac{\partial \mathcal{H}}{\partial X}\right]^T \tag{146}$$

$$\lambda_f = \left[\frac{\partial \Phi}{\partial X_f}\right]^T, \tag{147}$$

with

$$\frac{\partial \mathcal{H}}{\partial u} = 0, \qquad \frac{\partial \mathcal{H}}{\partial v} = 0, \tag{148}$$

or, if the control is bounded,

$$\mathcal{H}^* = \max_v \min_u \mathcal{H}. \tag{149}$$

The transversality condition may also apply,

$$\mathcal{H}_f + \frac{\partial \Phi}{\partial t_f} = 0. \tag{150}$$

Using the applicable set of necessary conditions, Horie and Conway [66] solved an optimal fighter pursuit-evasion game via two-sided optimization with a method they developed called the semi-direct collocation with nonlinear programming (SDCNLP). With this method, the optimal control for one player is found numerically where a GA is used an an initial guess [67] for the NLP solver and the optimal control for the other is found based on the analytic necessary conditions of the two-sided optimization problem. In this method, the adjoint variables for one player are included in the direct collocation NLP solver, and then the control variables for that player are found by using the solved costates in the equation resulting from applying Pontryagin's Minimum Principle. Thus, the direct collocation NLP to be solved is the system

described by both players' dynamics, their initial conditions, the original and modified boundary conditions taking into account the terminal costate conditions of one of the players, the control path constraints for both players, the costate dynamics of one player, and Pontryagin's Minimum Principle for one player, with the goal of maximizing (or minimizing depending on which player is chosen for the costates) the cost function which in their problem was of the Mayer form:

$$J = \phi(X_p(t_f), X_e(t_f), t_f). \tag{151}$$

Pontani and Conway [68] then took the SDCNLP method and developed an algorithm for the numerical solution of the three-dimensional orbital pursuit-evasion game where interception concludes the game once the instantaneous positions coincide. The objective of the pursuer is to minimize the interception time where the evader's objective is to maximize it. A low, constant thrust-to-mass ratio is assumed for both spacecraft and each begins maneuvering simultaneously possessing complete and instantaneous information about the state of the opposing player. The necessary conditions for existence of a saddle-point solution of a zero-sum differential game begin with introducing the Hamiltonian, which is separable,

$$\mathcal{H} = \lambda_p^T f_p + \lambda_e^T f_e, \tag{152}$$

and the terminal function $\Phi$, where the lengthy necessary conditions resulting in a TPVBP are reported in [68]. Pontani and Conway then used the SDCNLP method which as discussed transforms the two-sided optimization problem into a single-objective optimal control problem and transcribed it to an NLP with an initial guess provided by a GA like Horie and Conway.

Shen, et al. [69, 70] studied the pursuit-evasion orbital game where the pursuer

minimizes the time to interception while the evader attempts to maximize it for collision avoidance. They first had the pursuer rotate its orbit to the same plane as that of the evader and then solved the game via a MATLAB-based optimization environment called TOMLAB to obtain the optimal open-loop trajectories representing the saddle-point equilibrium solution.

Shen, et al. [71, 72] also used a pursuit-evasion game approach to deal with imperfect measurements and information with uncertainties for a satellite to track a GEO satellite where the entropy was to be minimized by the pursuer and maximized by the evader. The satellites both used continuous low-thrust with the control being the direction of thrust, and the worst case maneuvering strategies were obtained from the Nash equilibrium of the pursuit-evasion game by using fictitious play to solve the game problem. The cost function was the entropy,

$$J = \frac{1}{2}\ln((2\pi \exp)^6 \det(P)), \tag{153}$$

where $P$ is the error covariance matrix. The game was solved sequentially, where at each time step each player observes the actions of the other and responds with the best strategy. They furthered their work by using a similar approach where the controls of the pursuer were sensor resources which they used to minimize entropy while the evader attempted to maximize it by performing maneuvers where the same fictitious play concept was used to solve the problem [73].

Differential game solution techniques applied in the linearized relative frame for satellite motion include the work of Stupik, Pontani, and Conway [74, 75] where the objective of the pursuer is to minimize the time to capture and the objective of the evader is to maximize it. The control is the two angles of the constant low-thrust in three-dimensional space and mass loss is accounted for. The solution is then obtained by applying the analytic necessary and sufficient conditions for optimality and solving

the resulting system with a PSO where only three of the initial twelve costates are needed as the PSO optimization variables to determine the solution. This technique is basically the IHM applied to a pursuit-evasion game instead of to a one-sided optimization problem. Thus, the same conditions required to use the IHM must apply in order to use this solution technique.

Sun, et al. [76] calculated the numerical solution of a pursuit-evasion game of two spacecraft in low Earth orbit using the HCW equations via two methods they introduce: the semi-direct control parameterization (SDCP) method and a hybrid method which combines the SDCP method with the multiple shooting method. Instead of using the collocation method like the SDCNLP method, they used a control parameterization method which results in an approximate, smaller dimensional problem. Their methods involve solving two optimal control problems instead of one like in the SDCNLP method and they claim that it's equivalent to the original differential game unlike the SDCNLP method. The two optimal control problems can be solved via NLP solvers where a multi-objective genetic algorithm (MOGA) is used to find a good initial guess. For the hybrid method, they used the solution from the SDCP method as the initial guess for the multiple shooting method and increased the accuracy of the solution. In their examples they assumed a constant mass, however, and the cost was the separation distance between the two spacecraft at a given terminal time instead of the final time itself with capture terminal constraints.

Selvakumar and Bakolas [77] recently solved the pursuit-evasion game in Hill's frame, where the pursuer satellite aims to minimize time to capture while the evader attempts to prolong it. Both spacecraft had control constraints and the problem was solved by transforming the free final time problem into a fixed final time problem with a terminal cost by first transforming the relative motion equations into Levi-Civita coordinates [78], where the equations of motion become decoupled harmonic

oscillators, and then employing the Gutman, Esh, and Gefen state transformation [79]. A semi-analytical solution was presented for the time of capture and a closed-form expression was developed for the optimal control inputs. However, the problem was limited to the orbital plane and mass was assumed constant.

The research herein intends to build upon differential game solution techniques presented in this section to solve Problem C. Specifically, the research herein will expand upon the IHM and will use the Tschauner-Hempel equations of motion in order to account for any eccentricity in a potentially maneuvering RSO close to GEO, thus reducing the error which would accrue with the HCW equation of motion. Also, constant, steerable thrust will be used, which may be especially applicable for electric engines in such a game. It is desired to use the IHM due to the relative simplicity of the method and the fewer assumptions required compared to some of the other techniques. Thus, IHM will be used to formulate and solve multiple types of games (not just the intercept game), where each type corresponds to a different inspection goal of the pursuer.

## 2.8 Summary

This chapter was intended to provide sufficient background material to answer the research questions and expose the limits of previous work which this research is meant to extend. Different types of relative satellite dynamics models were introduced, along with their geometric parameterizations, and teardrop parameters were introduced, which will all be used to formulate and solve optimal control and differential game problems. A brief overview of optimal control and solution methodologies were provided, and previous work related to Problems A and B were presented. Regarding Problem A, a gap exists with respect to the combination of multiple aspects of previous efforts into one problem, accounting for a single, finite-thrust, slew rate lim-

ited engine and a keep-out cone, as well as the optimization of a combined formation establishment and reconfiguration problem. Regarding Problem B, a gap exists with respect to using a semi-analytic guidance method for on/off thrusters with various optimization algorithms to find the optimal finite burns to inject into and maintain teardrop trajectories (as well as NMCs), subject to lighting and collision constraints.

Metaheuristic techniques used in recent research to solve similar problems were also introduced, where a different type of control parameterization will be developed for Problem B in order to quickly use metaheuristic algorithms to generate initial guesses or complete solutions for Problem B. Also, the IHM was introduced, which will be used in Problem C to solve pursuit-evasion games, where more accurate equations of motions will be used and multiple game conditions will be solved.

Thus, this work will extend the previous work presented in this section by formulating and solving new problems, developing more efficient and reliable algorithms, accounting for more realistic control and constraints, and providing mission planners with options for solution tools and problem formulations based on mission requirements.

# III. Problem A

## 3.1 Overview

Problem A is a type of initial inspection of a non-maneuvering target satellite with the goal of visiting all eight octants surrounding the target. Specifically, this problem is a formation establishment and reconfiguration problem linked together into one multi-phase optimization problem, meaning for example that the optimal formation establishment solution may change if it sufficiently improves the reconfiguration solution. For Problem A, the inspector satellite has one, body-fixed engine with variable thrust and maximum slew rates. In addition to these unique control constraints, a keep-out cone is attached to the RSO, pointing towards nadir, which the inspector satellite may not enter. Given the mission, control type, and path constraint, the goal is to formulate and solve an optimal control problem to find minimum-time and minimum-fuel solutions from an arbitrary state nearby. The keep-out cone ends up defining two possible NMCs which the satellite may enter to stay outside of but come as close as possible to inspect the nadir-facing side of the target. Terminal constraints should allow the inspector to enter the NMCs at whichever point is optimal, and allow the optimizer to determine which of the two initial NMCs is optimal instead of defining which one to enter a priori. Two angles representing the thrust direction are introduced as additional states in order to limit slew rates to reasonable levels which needs consideration when using a low-thrust, body-fixed engine. Throttle is an additional control which allows the optimizer to determine when to thrust vs. coast, instead of defining the thrust-coast sequence(s) a priori. Due to the complexity of the problem, pseudospectral methods are used to obtain solutions. The goal is to first calculate the minimum-time solution and then solve increasing fixed final time problems while minimizing fuel to provide mission planners with optimal time vs.

fuel options. The research in this chapter has been published in [80].

The inspection mission is split into four phases. The goal of the first phase is to inject the satellite into an NMC about the target while avoiding the exclusion cone but coming as close as possible in order to inspect the nadir-facing side of the target. Instead of defining a specific NMC and injection point, this problem formulation will allow the inspector to inject itself at any point along one of two NMCs. The second phase is simply the resulting natural motion of the terminal conditions from phase one for a specified amount of minimum dwell time. The third phase starts at the terminal conditions of phase two, with the goal of transferring to an orthogonal NMC, i.e. a reconfiguration. The fourth phase is simply the resulting natural motion of the terminal conditions from phase three, resulting in the inspector having viewed the target from all eight octants surrounding the target provided by the two orthogonal NMCs. Since the motion in the two NMCs requires no control and the times in each NMC are specified by the user, the only phases explicitly included in the multi-phase optimization problem are phases one and three, which will be called such, even though phase three is in fact phase two of the optimization problem.

## 3.2 Equations of Motion and Control Definition

The equations used for Problem A are the HCW Equations, Equations 42–44, expressed in the LVLH frame (also called the RSW frame), as shown in Figure 3. The acceleration terms are added to the right hand side of the equations, resulting in

$$\ddot{x} - 2\omega\dot{y} - 3\omega^2 x = a_x \tag{154}$$

$$\ddot{y} + 2\omega\dot{x} = a_y \tag{155}$$

$$\ddot{z} + \omega^2 z = a_z, \tag{156}$$

where the three acceleration terms result from thrust forces acting on the inspector satellite in each of the three directions.

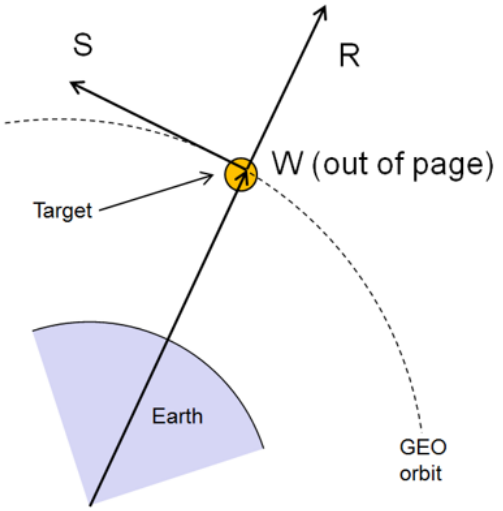



**Figure 3. LVLH or RSW Coord. Frame**

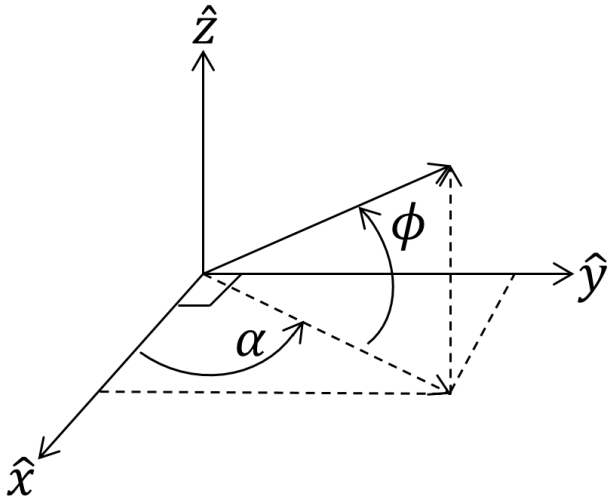



**Figure 4. Thrust Direction Variables**

The control type used in Problem A is continuous low-thrust control, where the thrust can be pointed in one direction only, e.g. one electric engine fixed to the satellite body frame. The rate at which the satellite and thus the thrust vector can slew is limited, which needs to be addressed for a body-fixed, low-thrust engine and helps provide realistic results. Thus, two additional states are added to define the direction of the thrust vector in 3-D space, $\alpha$ and $\phi$, which can be seen in Figure 4, where $\alpha$ is the in-plane angle and $\phi$ is the out-of-plane angle. The acceleration magnitude is allowed to vary, since this allows the optimizer to determine thrust and coast arcs, instead of the user picking a possibly suboptimal sequence a priori. As will be seen later, solutions are typically constant-magnitude, finite burn and coast sequences, thus allowing the acceleration magnitude to throttle may be used to determine finite-thrust and coast sequences for on/off thrusters. Finally, time-varying mass loss is accounted for. Given the defined control, the acceleration terms for Problem A can

be described by,

$$a_x(t) = T(t)a(t)\cos\phi(t)\cos\alpha(t) \tag{157}$$

$$a_y(t) = T(t)a(t)\cos\phi(t)\sin\alpha(t) \tag{158}$$

$$a_z(t) = T(t)a(t)\sin\phi(t), \tag{159}$$

where $T(t) \in [0, 1]$ is the throttle value, $a(t)$ is the maximum available acceleration magnitude which increases with mass loss, and $\alpha$ and $\phi$ are controlled by $\dot{\alpha}$ and $\dot{\phi}$ respectively, where $\dot{\alpha}, \dot{\phi} \in [-\delta, \delta]$ and $\delta$ is the maximum allowable slew rate. Note that the slew rate limit for $\dot{\alpha}$ is technically a non-inertial rate, since the RSW frame is rotating at about $\omega = 7.3X10^{-5}$ rad/s at GEO. This minimal value is negligible compared to a reasonable maximum slew rate value of, for example, 0.5 deg/s. However, this would need to be taken into account for low-Earth orbits. In summary, there are eight states, $X = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \alpha, \phi]$, and three controls, $u = [T, \dot{\alpha}, \dot{\phi}]$. Mass loss is accounted for by using the following model for the acceleration magnitude at any point in time where the thrust, mass flow rate, and effective exhaust velocity are assumed constant,

$$a(t) = \frac{a_0}{1 - t\frac{a_0}{c}}, \tag{160}$$

where $a_0$ is the initial acceleration due to the constant thrust and initial mass, $c$ is the constant effective exhaust velocity, and $t$ is the elapsed time the thruster has been on since $t_0$.

### 3.3 Targeted NMCs

The RSO is assumed to be pointing instruments or communication devices towards nadir, and thus an exclusion cone is projected from the RSO towards nadir, oriented

along the $-\hat{x}$ axis, and stationary in the LVLH frame, which the inspector cannot enter so as to not interfere with the satellite's operations. This exclusion cone thus imposes constraints on the trajectory the satellite can take during proximity operations, and also defines the angle at which the NMC needs to be tilted with respect to the $xy$ plane to keep out of the exclusion cone, but stay as close as possible to during the initial NMC. This exclusion cone also constrains the trajectory the satellite can take when transferring to the second NMC after sufficient information has been collected in the initial NMC. Figure 5 shows the options for the first NMC and the second, orthogonal NMC which is chosen based on the first NMC.

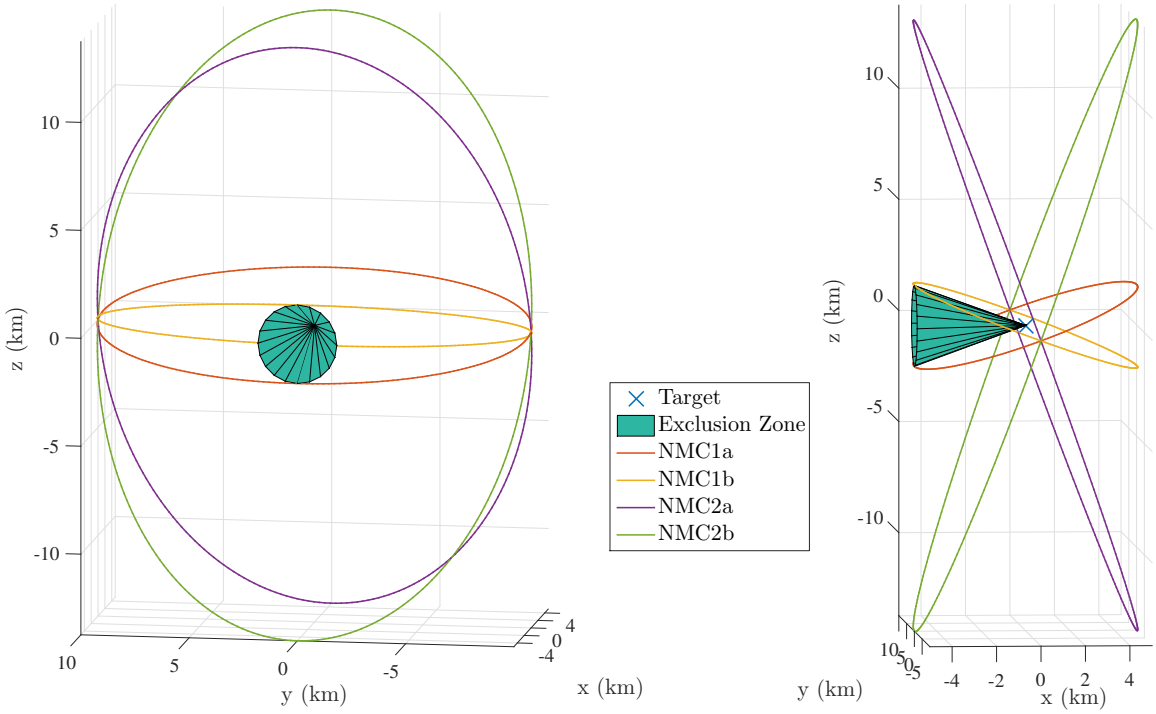



**Figure 5. Exclusion Cone and NMC Options**

The natural, unforced trajectory of the inspector with respect to the target depends on the initial unforced conditions of the inspector in the relative frame upon completion of the maneuver. The resulting trajectories generally drift over time in the LVLH frame, distancing themselves from the target. However, the one condition

which ensures a zero-drift relative trajectory is Equation 82, which generates a $2 \times 1$ ellipse in the orbital plane of the RSO. Given this condition, what remains to be defined is the offset of the ellipse in the $y$ direction, the size of the NMC in the $xy$ plane, the tilt with respect to the $xy$ plane or the maximum out-of-plane amplification, and the phase relationship between the in-plane and out-of-plane motion. Using the LROEs, the zero-drift condition makes $x_d = 0$ and $y_d = y_{d_0}$. To make the RSO be at the center of the NMC, $y_d$ must be set to zero by the constraint:

$$y\omega = 2\dot{x}. \tag{161}$$

By choosing a value for $a_e$, a terminal constraint can be set on the states to obtain the desired size of the ellipse:

$$\frac{a_e^2}{4} = \left(\frac{\dot{x}}{\omega}\right)^2 + \left(3x + 2\frac{\dot{y}}{\omega}\right)^2. \tag{162}$$

Then, using the LROEs with the constant phase difference, $\gamma = \psi - \beta$, between the periodic motion in the orbit plane and the periodic motion in the $z$ direction, the six relative states can be described in terms of the LROEs as the following, stated again here for ease of reference:

$$x = -\frac{a_e}{2}\cos\beta + x_d \tag{163}$$

$$y = a_e \sin\beta + y_d \tag{164}$$

$$z = z_{max}\sin(\gamma + \beta) \tag{165}$$

$$\dot{x} = \frac{a_e}{2}\omega\sin\beta \tag{166}$$

$$\dot{y} = a_e\omega\cos\beta - \frac{3}{2}\omega x_d \tag{167}$$

$$\dot{z} = z_{max}\omega\cos(\gamma + \beta). \tag{168}$$

For this problem, due to the exclusion cone, $\gamma$ should be $\pm 90^o$ in order for the cross-track motion to reach its maximum (or minimum value) when it reaches the minimum radial distance from the RSO, and be outside of but as near as possible to the exclusion cone. It follows that

$$z = z_{max}(\pm \cos \beta), \tag{169}$$

and with $x_d = 0$,

$$\cos \beta = -\frac{2x}{a_e}, \tag{170}$$

and thus

$$z = z_{max}\left(\pm\left(-\frac{2x}{a_e}\right)\right) = \mp z_{max}\frac{2x}{a_e}. \tag{171}$$

For the optimizer to be able to converge on either of the two initial NMCs shown in Figure 5, Equation 171 is then squared. Making this a terminal constraint allows the optimizer to converge to either of the two ellipses. To ensure it's also moving in the correct direction at the terminal time, its first derivative also becomes a terminal constraint. In summary, the following two equations become terminal constraints:

$$z^2 = z_{max}^2\frac{4x^2}{a_e^2} \tag{172}$$

$$2z\dot{z} = z_{max}^2\frac{8x\dot{x}}{a_e^2}. \tag{173}$$

Thus, using the constraints presented in this section, the user simply defines $a_e$ and $z_{max}$, where the latter comes from the given exclusion cone half-cone angle, $\theta$. Given these parameters, the desired NMCs around the exclusion cone are defined.

## 3.4 Optimization Problem Formulation

This section describes the optimization problem formulation, organized by phases. For each phase, the equations of motion and control variables are summarized, and any applicable state or control bounds are presented. Applicable and non-trivial constraints on the initial and terminal conditions, the path, and linkage constraints to the next phase are also presented. Finally, the multi-phase performance index is discussed.

### 3.4.1 Phase One: Formation Establishment.

As discussed, this problem is split into four different phases of motion, with phase one and three tied together into one multi-phase optimization problem. The first phase starts with the inspector nearby in the LVLH frame, i.e., its arbitrary starting location is close enough to the target for the HCW equations to apply. The goal of the first phase is to inject the inspector into one of the two NMCs which touch the top or bottom of the exclusion cone, while minimizing the multi-phase problem performance index. The equations of motion for phase one (and phase three) can be condensed to the following:

$$\ddot{x}(t) - 2\omega\dot{y}(t) - 3\omega^2 x(t) = T(t)a(t)\cos\phi(t)\cos\alpha(t) \tag{174}$$

$$\ddot{y}(t) + 2\omega\dot{x}(t) = T(t)a(t)\cos\phi(t)\sin\alpha(t) \tag{175}$$

$$\ddot{z}(t) + \omega^2 z(t) = T(t)a(t)\sin\phi(t), \tag{176}$$

where the control variables are $T$, $\dot{\alpha}$, and $\dot{\phi}$ as previously discussed. There are no bounds on the states (aside from those limiting the search space of the NLP solver) which should be active. Bounds on the angle rates limit the rate at which the satellite

can slew, and are the same for phases one and three:

$$-\delta \leq \dot{\alpha}, \dot{\phi} \leq \delta. \tag{177}$$

The throttling percentage is limited for both phases one and three as well:

$$0 \leq T \leq 1. \tag{178}$$

The initial conditions on the states are fixed to arbitrary values, whereas the terminal conditions on the states are free but constrained to inject the inspector satellite into either of the two NMCs just touching the top or bottom of the exclusion cone. Thus, phase one terminal constraints are Equations 82, 161, 162, 172, and 173 shown again here for convenience, and where the superscripts in parenthesis denote the phase for that specific variable:

$$\dot{y}_f^{(1)} = -2\omega x_f^{(1)} \tag{179}$$

$$y_f^{(1)}\omega = 2\dot{x}_f^{(1)} \tag{180}$$

$$\frac{a_e^2}{4} = \left(\frac{\dot{x}_f^{(1)}}{\omega}\right)^2 + \left(3x_f^{(1)} + 2\frac{\dot{y}_f^{(1)}}{\omega}\right)^2 \tag{181}$$

$$z_f^{2(1)} = z_{max}^{2(1)} \frac{4x_f^{2(1)}}{a_e^2} \tag{182}$$

$$2z_f^{(1)}\dot{z}_f^{(1)} = z_{max}^{2(1)} \frac{8x_f^{(1)}\dot{x}_f^{(1)}}{a_e^2}. \tag{183}$$

There are no path constraints (than otherwise noted state and control bounds), and linkage constraints ensure that the final conditions in phase one are the initial conditions in phase three propagated backwards analytically by the minimum time the inspector satellite must spend in the initial NMC, $t_T$. Using the HCW STM, Equation

51, but propagating backwards by $t_T$, the linkage constraints are

$$\begin{bmatrix} x_f \\ y_f \\ z_f \\ \dot{x}_f \\ \dot{y}_f \\ \dot{z}_f \end{bmatrix}^{(1)} = \Theta(-t_T, 0) \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \end{bmatrix}^{(3)} \tag{184}$$

$$t_f^{(1)} = t_0^{(3)}, \tag{185}$$

where the times between phases one and three can be linked directly since those are the two phases being optimized.

### 3.4.2 Phase Two: Initial NMC.

Phase two is the natural, unforced motion due to the terminal conditions from phase one. The motion is a 2x1, zero-drift NMC which just touches either the top or bottom of the exclusion cone. The dynamics are simply the left hand side of Equations 174–176 set equal to zero and are propagated forward analytically for the desired minimum dwell time in the first NMC, $t_T$. Thus, this phase is implicitly contained in the overall problem via linkage constraints but not an explicit phase of the optimization problem.

### 3.4.3 Phase Three: Formation Reconfiguration.

Phase three is the transfer from the initial NMC to the orthogonal NMC with the goal of obtaining spherical coverage of the target. Due to the nature of the control, if it's optimal to remain in the initial NMC, i.e., have $T = 0$ for a period of time, then it may do so. Phase three then uses the same dynamics and control as in Equations

174–176 to transfer to an orthogonal ellipse. State and control bounds are the same as in phase one, and the initial conditions are the final conditions from phase two. The terminal constraints must force the inspector to maneuver into an orthogonal ellipse, which is defined based off of the ellipse the optimizer converged on in phase one. Thus, the algorithm compares $x_f^{(1)}$ and $z_f^{(1)}$ to determine whether they are in or out of phase, and defines the orthogonal ellipse to be the opposite, where $p_3$ is defined to be $-1$ or 1 respectively. Thus, squaring Equation 171 is no longer needed, and the new maximum cross-track amplification for phase three, $z_{max}^{(3)}$, is determined by a pseudo half-cone angle, $\tilde{\theta} = 90^o - \theta$. Thus, the terminal constraints for phase three are

$$\dot{y}_f^{(3)} = -2\omega x_f^{(3)} \tag{186}$$

$$y_f^{(3)}\omega = 2\dot{x}_f^{(3)} \tag{187}$$

$$\frac{a_e^2}{4} = \left(\frac{\dot{x}_f^{(3)}}{\omega}\right)^2 + \left(3x_f^{(3)} + 2\frac{\dot{y}_f^{(3)}}{\omega}\right)^2 \tag{188}$$

$$z_f^{(3)} = p_3 z_{max}^{(3)} \left(\frac{-2x_f^{(3)}}{a_e}\right). \tag{189}$$

Path constraints for phase three ensure the inspector does not enter the exclusion cone,

$$\frac{\left[x(t)^{(3)}, y(t)^{(3)}, z(t)^{(3)}\right][-1,0,0]^T}{||[x(t)^{(3)}, y(t)^{(3)}, z(t)^{(3)}]||_2} = \frac{-x(t)^{(3)}}{\sqrt{x^2(t)^{(3)} + y^2(t)^{(3)} + z^2(t)^{(3)}}} \leq \cos\theta, \tag{190}$$

where the cone is aligned with the $-\hat{x}$ axis pointing towards nadir. Phase three is the last phase included in the optimization problem, and its final time is either free or fixed, depending on the performance index for the multiple phase optimization problem.

### 3.4.4 Phase Four: Orthogonal NMC.

Phase four is the natural, unforced motion due to the terminal conditions from phase three. The resulting NMC is orthogonal to the initial one, resulting in spherical coverage of the target. It is not included as part of the multiple phase optimization problem and is analytically propagated forward from the terminal conditions of phase three.

### 3.4.5 Performance Indices.

The optimization problem is first solved by minimizing the overall final time, i.e., the performance index is

$$J = t_f^{(3)}, \tag{191}$$

where $t_f^{(3)}$ is the total time it takes phases one and three combined, not just phase three. In order to help reduce control chatter and produce more acceptable results, $\dot{\alpha}^2$ and $\dot{\phi}^2$ are also minimized during both phases and multiplied by a relatively small weight, $W$, so that their effect on the minimum-time solution is negligible:

$$J = t_f^{(3)} + W \int_{t_0^{(1)}}^{t_f^{(3)}} \left( \dot{\alpha}^{2(1)} + \dot{\phi}^{2(1)} + \dot{\alpha}^{2(3)} + \dot{\phi}^{2(3)} \right) \mathrm{d}t. \tag{192}$$

Using this as a baseline for mission design, the problem is then solved by minimizing the fuel for multiple fixed final time problems, where $t_f^{(3)}$ is fixed and chosen to be values increasing from the minimum-time solution. For these problems,

$$J = \int_{t_0^{(1)}}^{t_f^{(3)}} \left[ \left( T^{(1)} + T^{(3)} \right) + W \left( \dot{\alpha}^{2(1)} + \dot{\phi}^{2(1)} + \dot{\alpha}^{2(3)} + \dot{\phi}^{2(3)} \right) \right] \mathrm{d}t. \tag{193}$$

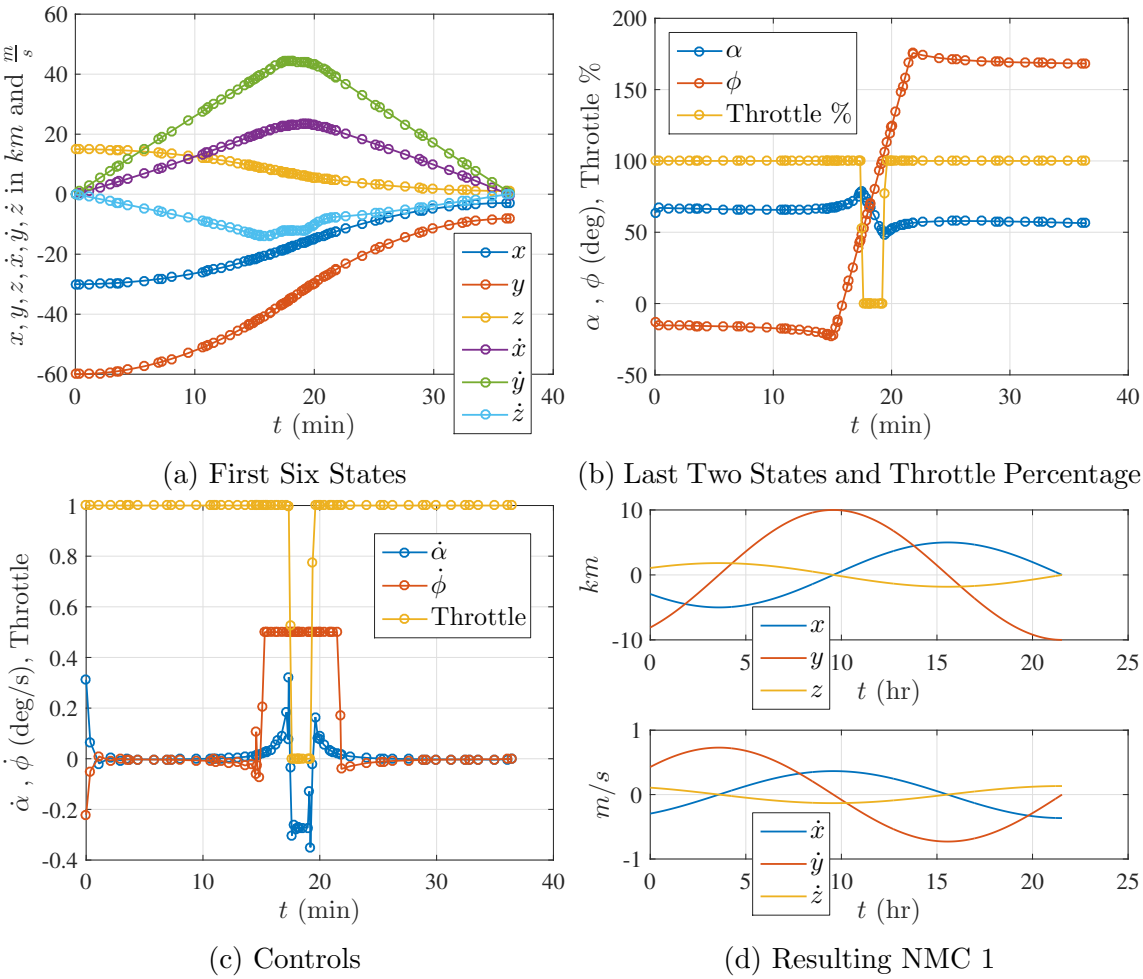
### 3.5 Simulation and Results

Simulations were run with the parameters shown in Table 2, where $\alpha_0^{(1)}$ and $\phi_0^{(1)}$ were chosen to be initially thrusting the inspector towards the target.

**Table 2. Problem A Simulation Parameters**

| Spacecraft Properties | Initial Position | Initial Velocity | Initial Angles | NMC Parameters |
|---|---|---|---|---|
| $a_0 = 0.05$ N/kg | $x_0^{(1)} = -30$ km | $\dot{x}_0^{(1)} = 0$ m/s | $\alpha_0^{(1)} = 63.43^o$ | $a_e = 10$ km |
| $c = 3.33$ km/s | $y_0^{(1)} = -60$ km | $\dot{y}_0^{(1)} = 0$ m/s | $\phi_0^{(1)} = -12.60^o$ | $\theta = 20^o$ |
| $\delta = 0.5$ deg/s | $z_0^{(1)} = 15$ km | $\dot{z}_0^{(1)} = 0$ m/s | | |

The first solution presented is the minimum-time solution as seen in Figures 6, 7, and 8 where the circles represent the collocation points. For this particular run, the minimum required time for the first NMC is $t_T = 0.9P$, where $P$ is the period of the reference orbit. The minimum-time solution for the parameters in Table 2 is 36.78 minutes where phase one takes 36.41 minutes and enters the initial NMC such that after $t_T$ has passed the inspector satellite's position is just shy of the crossing between the two NMCs, where phase three then only takes 0.38 minutes to transfer the satellite into the orthogonal NMC. The total fuel used, measured by the total integral of $T(t)$, can be simply thought of as the total time the thruster is on, $t_{on}$. For this minimum-time solution, $t_{on} = 34.91$ minutes, which is less than the minimum time of the maneuver since it's optimal for the thruster to turn off for a moment, as seen in Figure 6 (b) and (c), while the satellite turns itself around to start slowing it down. The short transfer from the initial NMC to the orthogonal one is practically a constant direction and constant magnitude burn, where $\alpha_0^{(3)}$ and $\phi_0^{(3)}$ are free to be optimized, i.e. it is assumed the satellite can orient itself to those rotational angles during the initial NMC to prepare for the burn. This tends to be a common optimal solution for different initial conditions and multiple minimum-time problems. That is, the first phase solutions change so that at the end of $t_T$, the phase three burns are




(a) First Six States

(b) Last Two States and Throttle Percentage

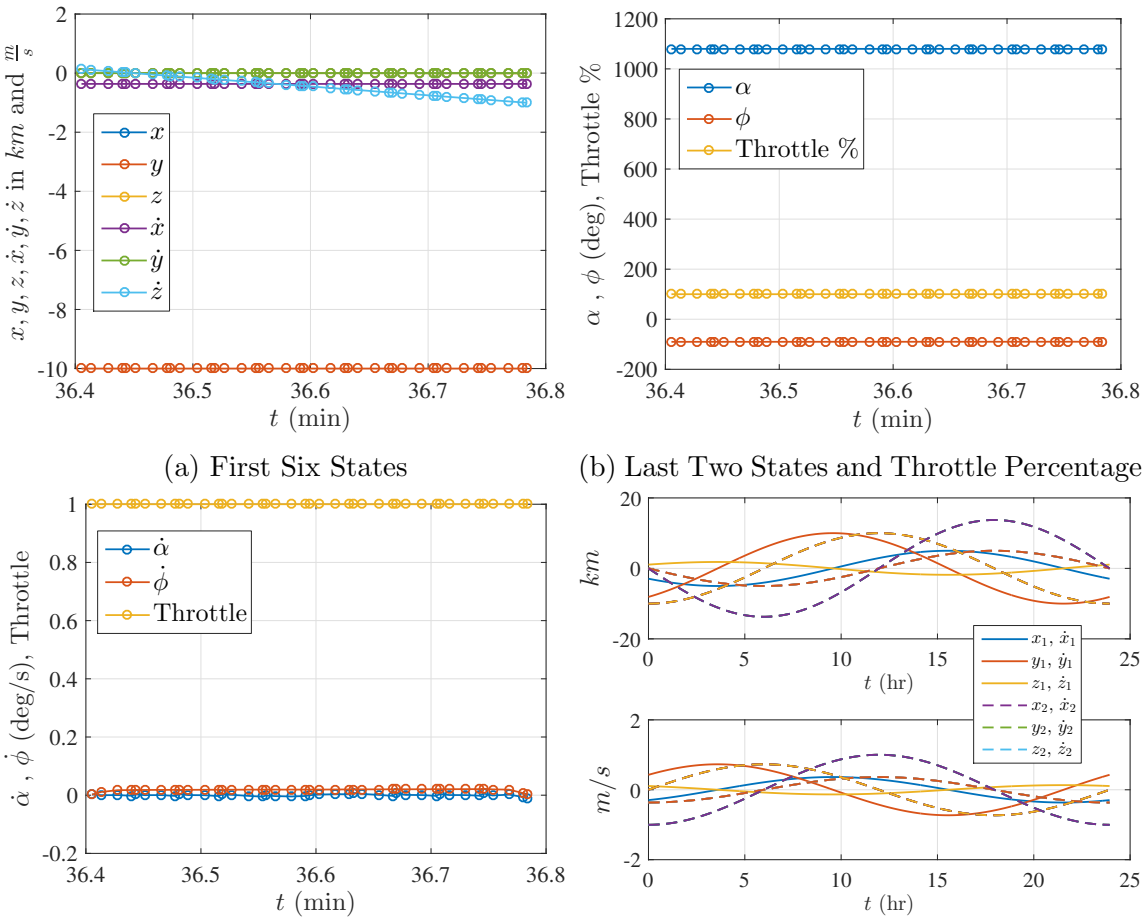
(c) Controls

(d) Resulting NMC 1

**Figure 6. Problem A, Min Time Solution, Phase 1, 90% of NMC 1**

very similar to that seen in this case. This is the result of combining the formation establishment and reconfiguration maneuvers into one optimization algorithm, and ends up saving time and fuel.

To show an example of how the inspector satellite may enter either of the two possible initial NMCs as shown in Figure 5, the exact same combined optimization problem is run as described by Table 2 and shown in Figure 8 but with $z_0^{(1)} = -15$ km instead of 15 km. Figure 9 shows how with this difference in initial conditions, the maneuver trajectory converges to the initial NMC which touches the bottom of the cone instead of the top of the cone, and the orthogonal NMC is subsequently




(a) First Six States

(b) Last Two States and Throttle Percentage

(c) Controls

(d) Resulting NMC 1 and 2 Motion

**Figure 7. Problem A, Min Time Solution, Phase 3, 90% of NMC 1**

changed.

To show how it does save time and fuel by combining the maneuvers into one optimization problem, the problem has also been solved where the two maneuvers are optimized separately. The minimum-time solution for phase one is 36.28 minutes as seen in Figure 10, which is less than the 36.41 minutes phase one solution for the combined optimization problem. However, the phase three minimum-time solution is 7.56 minutes as seen in Figure 11, which is much greater than the 0.38 minutes it takes phase three in the combined optimization problem. This leads to a total time to complete phases one and three of 43.84 minutes, approximately 19% more than the

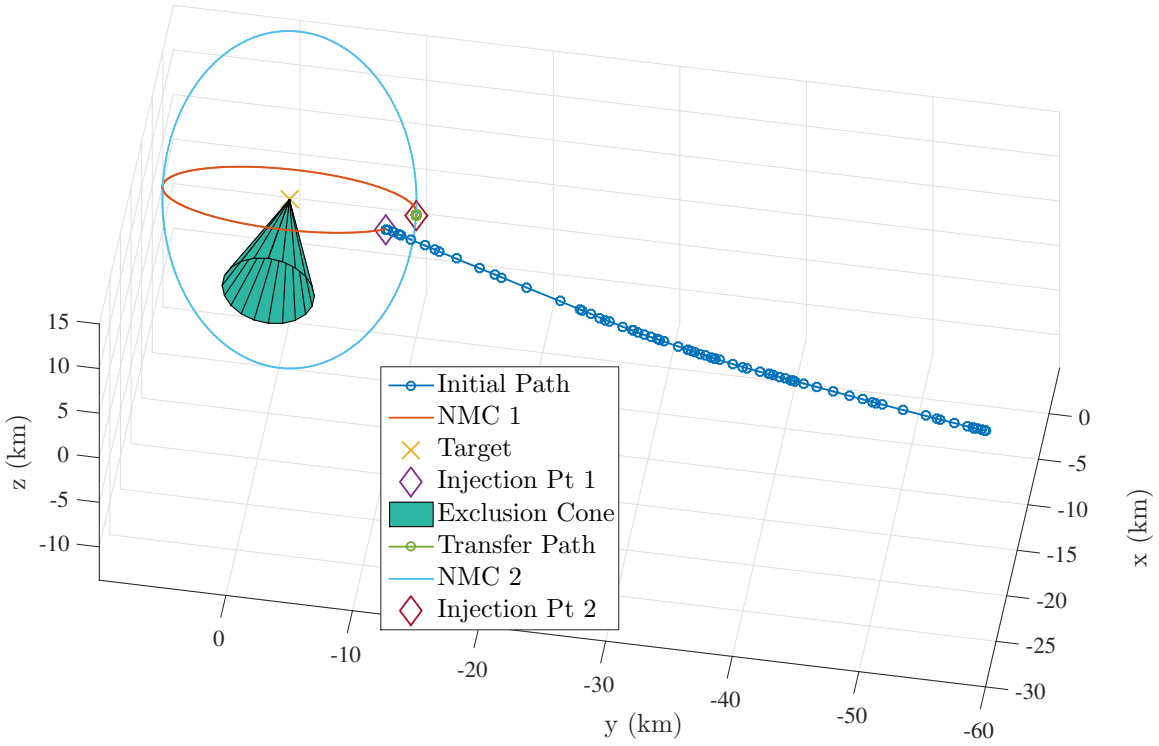



**Figure 8. Problem A, Min Time Solution Trajectory, 90% of NMC 1**

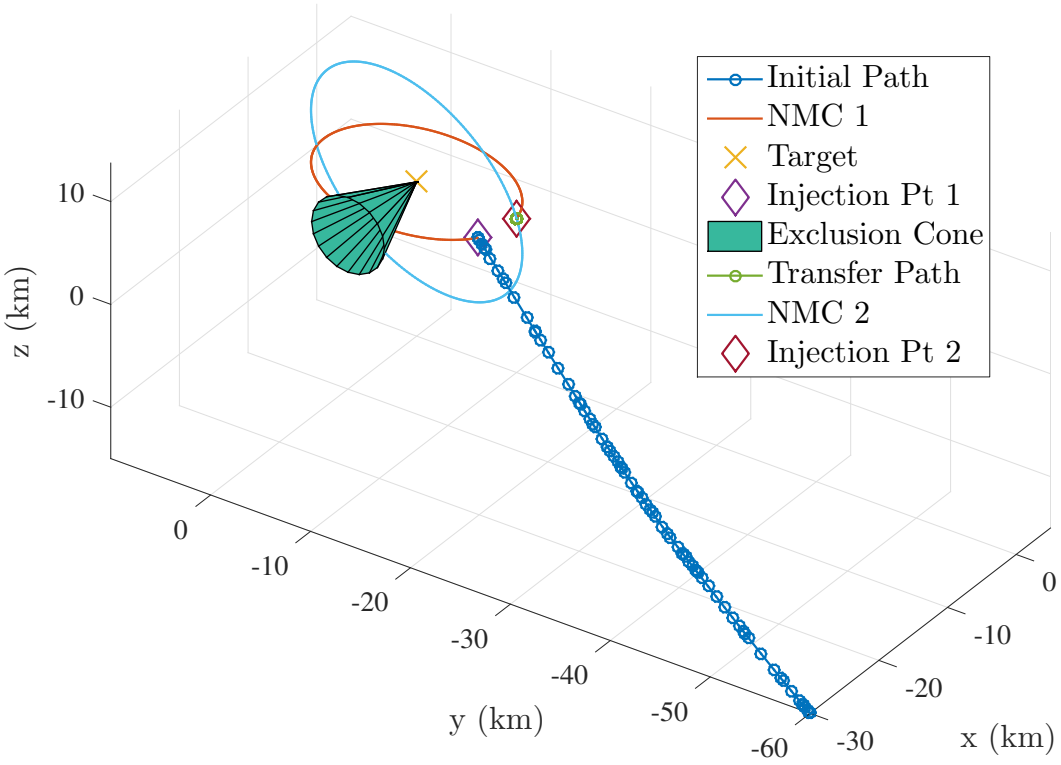
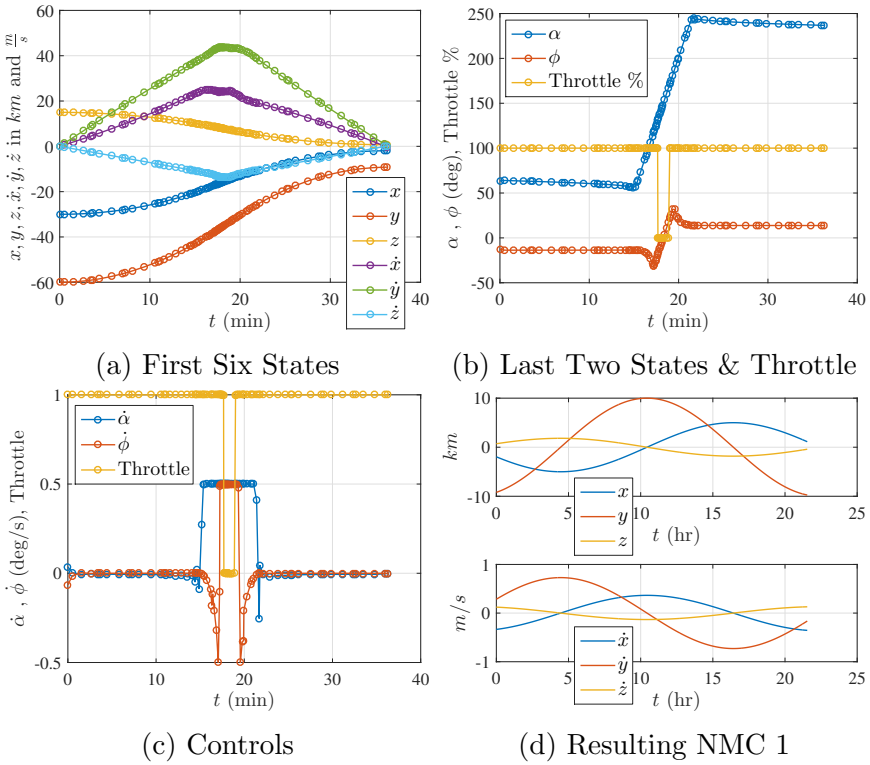



**Figure 9. Problem A, Min Time Solution Trajectory, 90% of NMC 1, $z_0^{(1)} = -15$ km**

combined optimization problem time of 36.78 minutes. The separated optimization problems also use a total of 7% more fuel than the combined problem, with $t_{on} = 37.33$ minutes as compared to 34.91 minutes. These time and fuel savings are especially important for minimum-time solutions, which savings can increase greatly if at the end of $t_T$ for the separate problems the inspector is far from the crossing of the two NMCs.

With the minimum-time solution as the baseline, the simulation was run multiple times for varying fixed final times, increasing the fixed final times from the minimum-time solution. The simulation was performed with $t_T = P$. Figures 12 and 13 show the minimum-fuel solution for the case where $t_f^{(3)}$ is fixed to 60 minutes. Phase one takes 59.62 minutes and phase two takes 0.38 minutes, and the combined formation establishment and reconfiguration maneuvers only require $t_{on} = 13.09$ minutes. A summary of the simulation results presented in this section are shown in Table 3. For these simulations, IPOPT was used as the NLP solver, the default (or tighter) GPOPS-II tolerances were used, and no special treatment regarding the settings were required to obtain the solutions.

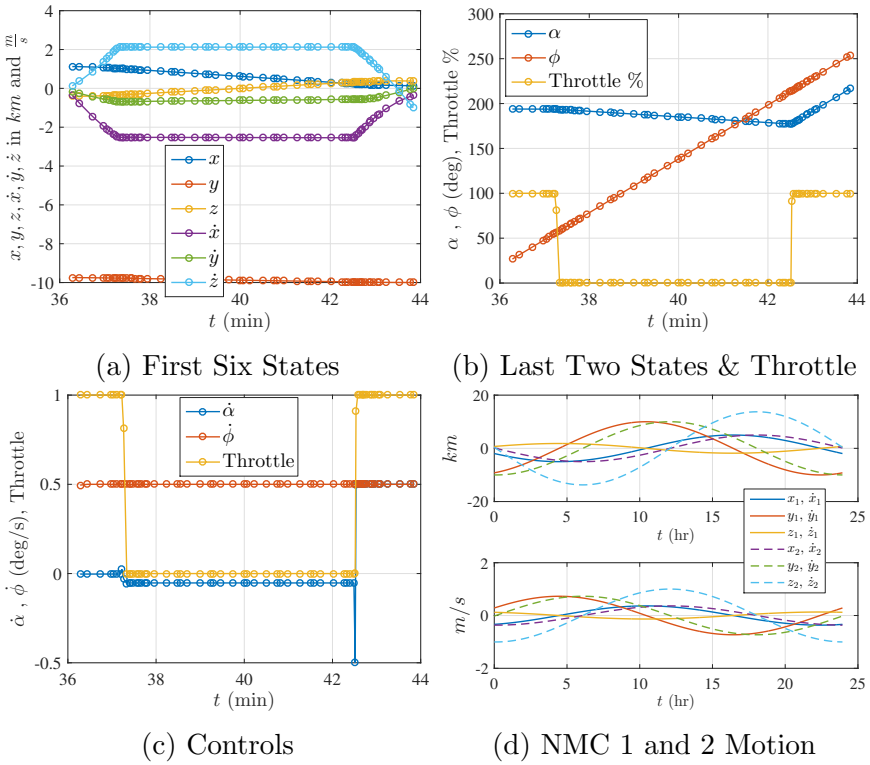
Figure 14 shows multiple minimum-fuel solutions, which shows that if the mission planner can provide more total time for phase one and phase three maneuvers to be completed, then fuel can be saved. Figure 14 also shows how minimizing the angular rate control chatter with a small weight as part of the cost function has negligible impact on the main goal of the optimization, which is to minimize fuel, since the cost and integral curves are practically indistinguishable.




(a) First Six States (b) Last Two States & Throttle

(c) Controls (d) Resulting NMC 1

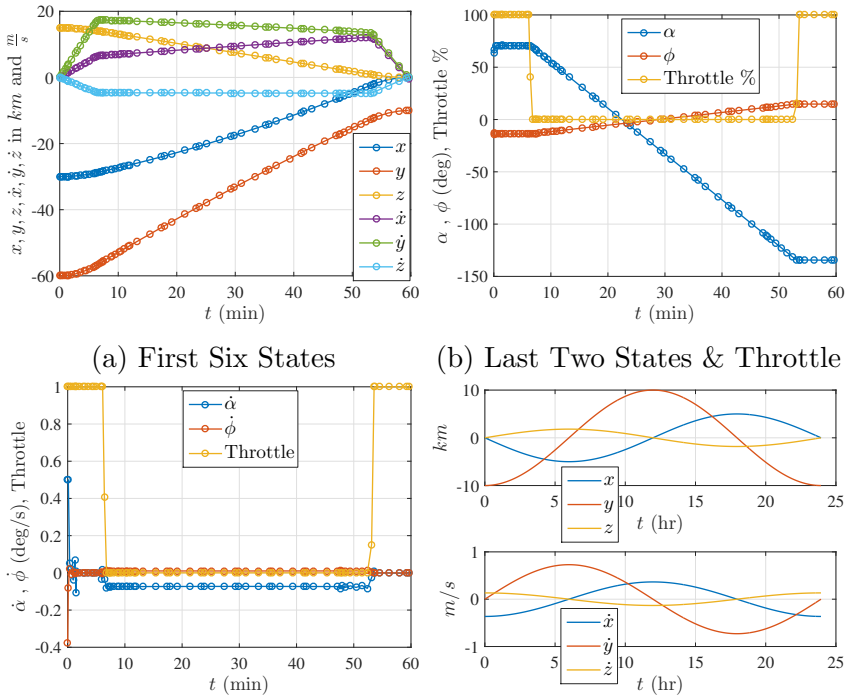
**Figure 10. Problem A, Min Time Solution, Phase 1 Optimized Alone, 90% of NMC 1**




(a) First Six States (b) Last Two States & Throttle

(c) Controls (d) NMC 1 and 2 Motion

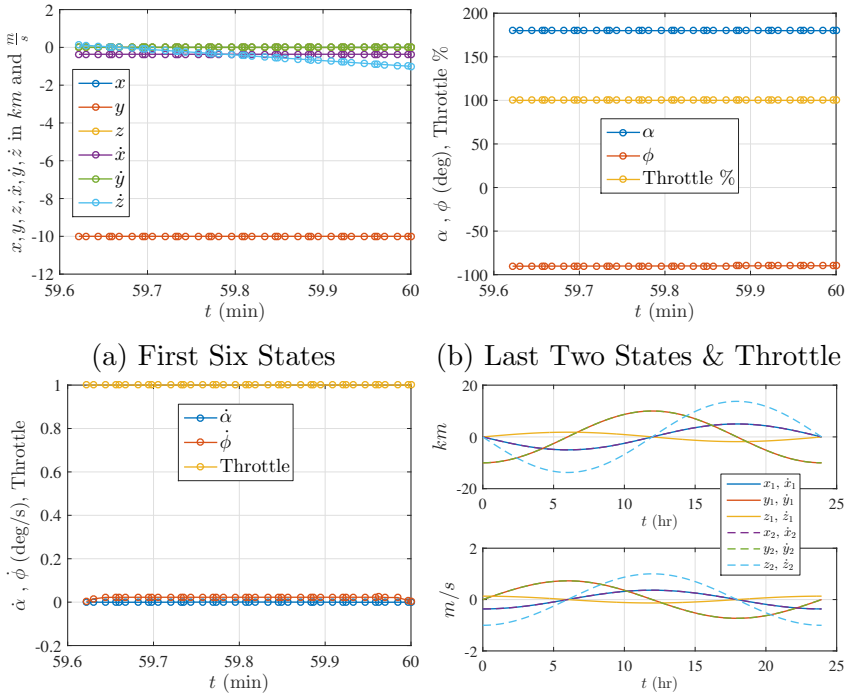
**Figure 11. Problem A, Min Time Solution, Phase 3 Optimized Alone, 90% of NMC 1**




(a) First Six States (b) Last Two States & Throttle

(c) Controls (d) Resulting NMC 1

**Figure 12. Problem A, Min Fuel Solution, Phase 1, 100% of NMC 1, $t_f^{(3)} = 60$ min**




(a) First Six States (b) Last Two States & Throttle

(c) Controls (d) NMC 1 and 2 Motion

**Figure 13. Problem A, Min Fuel Solution, Phase 3, 100% of NMC 1, $t_f^{(3)} = 60$ min**

**Table 3. Problem A Simulation Results (in minutes)**

| | Min time 90% Initial NMC | Min time 90% Initial NMC not combined | Min time 90% Initial NMC $z_0^{(1)} = -15$ km | Min throttle 100% Initial NMC $t_f^{(3)} = 60$ min |
|---|---|---|---|---|
| $t_f^{(1)}$ | 36.41 | 36.28 | 36.41 | 59.62 |
| $t_f^{(3)} - t_f^{(1)}$ | 0.38 | 7.56 | 0.38 | 0.38 |
| $t_f^{(3)}$ | 36.78 | 43.84 | 36.79 | 60.00 |
| $\int_{t_0}^{t_f^{(1)}} T(t)dt$ | 34.54 | 35.01 | 34.42 | 12.71 |
| $\int_{t_0}^{t_f^{(3)}} T(t)dt$ | 0.38 | 2.32 | 0.38 | 0.38 |
| $\int_{t_0}^{t_f^{(3)}} T(t)dt$ | 34.91 | 37.33 | 34.79 | 13.09 |
| $J$ | 36.78 | 43.84 | 36.79 | 13.09 |

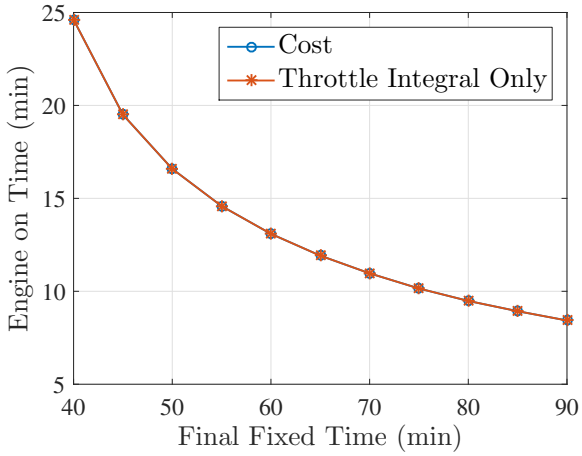



**Figure 14. Problem A, Range of Optimal Solutions, 100% of NMC 1**

## 3.6 Problem A Conclusion

The presented method to formulate and solve Problem A via pseudospectral methods provides a novel multi-phase approach to generate time-optimal and fuel-optimal maneuvers for a satellite with a unique control type to perform an initial inspection of an RSO with an exclusion cone in GEO. Regarding the unique control type, a continuous, low-thrust, body-fixed engine with slew rate limits and a maximum acceleration magnitude which increases with mass loss can be successfully incorporated into a complex, multi-phase optimal control problem. By introducing two angles as addi-

tional states, controlled by their respective and bounded angle rates, the translational trajectory can be successfully coupled to and constrained by rotational constraints. Also, instead of defining (possibly sub-optimally) thrust and coast sequences a priori, throttle can be used as a control variable to determine the sequence. Regarding the cost function, if a second, weighted term is introduced into the cost function for both the minimum-time and minimum-fuel problems, unwanted control chatter (especially during coast arcs) can be successfully minimized without affecting the actual goal of the cost function. Also, if the user knows the minimum amount of time desired to dwell in the initial NMC, then the optimization of the combined formation establishment and reconfiguration can save potentially significant time and/or fuel as compared to optimizing the formation establishment and reconfiguration maneuvers separately. Regarding the constraints, it is possible to formulate them such that the inspector satellite can converge to one of two initial NMCs formed about an exclusion cone, and is free to enter the the trajectory at any point along the NMC, which further decreases the performance index. Finally, the general architecture of the developed algorithm allows for the production of a family of solutions, where the Pareto front for a specific maneuver can be constructed to give mission planners choices based on time and fuel requirements. Thus, this algorithm produces optimal multi-phase mission planning, autonomously determining variables and incorporating realistic constraints which may have been difficult for mission planners to account for optimally beforehand.

# IV. Problem B

## 4.1 Overview

Problem B differs from Problem A in several ways. First, the inspector satellite is assumed to have on/off, finite thrusters and be capable of reorienting its thrust vector instantaneously. This means that the satellite most likely has multiple thrusters on multiple faces, where each thruster can either be on or off and has no throttling capability. It is assumed that for each burn, these thrusters combine in the most efficient manner to produce a known acceleration magnitude, where the direction of the acceleration vector is held constant in the relative frame for each burn. Problem B investigates how to use these constant magnitude, constant direction burns in sequences with coast phases to target certain relative trajectories. Another way in which Problem B differs from Problem A is that the trajectories targeted include teardrop trajectories, which are unbounded in the relative frame. Also, Problem B maneuvers are subject to various inspection constraints, including lighting and collision constraints.

This chapter is split up into three sections, denoted B-1, B-2, and B-3. Each section varies with respect to the type of control sequence used, the trajectory targeted, and the constraints imposed. Problem B-1 uses a burn-burn and a burn-coast-burn sequence and addresses optimal maneuvers into a teardrop trajectory. Problem B-2 also uses a burn-coast-burn sequence but targets NMCs subject to lighting constraints. Problem B-3 then investigates a coast-burn-coast-burn sequence, and addresses optimal maneuvers into a teardrop again, but now subject to the same constraints from Problem B-2 as well as to passive and active collision avoidance constraints. The research for B-1 has been published in [81] and [82], the research for B-2 has been published in [83] and [84], and the research for B-3 has been published in [85].

## 4.2 Problem B-1

### 4.2.1 Overview.

Depending on mission requirements, one type of inspection mission may require the inspector satellite to hover at a fixed point with respect to the RSO, either below or above the satellite in the radial direction. This type of hovering would require continuous thrusting, unless the fixed point is expanded to a keep-in volume in which a relative teardrop pattern is allowed to exist. With a relative teardrop trajectory, the inspector satellite can be injected into a trajectory which naturally forms the prescribed teardrop with respect to the RSO, and a small burn can be made to repeat the teardrop motion. This allows the satellite to perform mission operations during the natural motion portion of the teardrop, and can be called a quasi-hover. These types of trajectories may be desirable based on mission requirements. Thus, the goal of Problem B-1 is to generate optimal guidance for an inspector satellite to maneuver into and maintain a prescribed relative teardrop trajectory in three dimensions, in order to successfully inspect and characterize an RSO. The problem is solved using a burn-burn sequence for the minimum-time formulation and a burn-coast-burn sequence for the minimum-fuel formulation. These problems are solved by parameterizing the control, such that the optimal control problem is adequately represented by a static, or parameter, optimization problem. The initial guess is generated by using MATLAB's PSO, which is then used as the initial guess for both a mid and high-fidelity solution, where the mid-fidelity solution is calculated using MATLAB's *fmincon* and the high-fidelity solution is calculated using GPOPS-II.

### 4.2.2 Equations of Motion and Control Definition.

Problem B-1 uses the HCW equations of motion where for each burn the acceleration terms are constants, defined by a constant acceleration magnitude, and constant

in-plane and out-of-plane angles,

$$a_x = a_0 \cos\phi \cos\alpha \tag{194}$$

$$a_y = a_0 \cos\phi \sin\alpha \tag{195}$$

$$a_z = a_0 \sin\phi, \tag{196}$$

where $a_0$ is the initial acceleration magnitude due to a constant thrust and satellite mass at the time the burn starts, and where $\alpha$ and $\phi$ have been shown in Figure 4.

### 4.2.3 Analytic Propagation of a Constant Magnitude, Constant Direction Burn.

This subsection shows how there is an analytic solution to a constant acceleration magnitude, constant direction burn when using the HCW equations, which will then allow the optimal control problem to be posed as a parameter optimization problem where no discretization or numerical integration is required.

The HCW equations with the acceleration terms equal to Equations 194–196 have an analytic solution. Using MATLAB's *dsolve* function, the solution is,

$$\begin{aligned} x(d_b) = \frac{1}{4\omega^2}\Big(&4a_x + 8c_6\omega + 4a_x\cos(d_b\omega) + 8a_y\sin(d_b\omega) \\ &+ 8a_y d_b\omega - 3c_5\omega\cos(d_b\omega) + 3c_4\omega\sin(d_b\omega)\Big) \end{aligned} \tag{197}$$

$$\begin{aligned} y(d_b) = -\frac{1}{2\omega^2}\Big(&6c_1\omega^2 - 8a_y\cos(d_b\omega) - 8a_y + 4a_x\sin(d_b\omega) + 3a_y d_b^2\omega^2 \\ &+ 4a_x d_b\omega - 3c_4\omega\cos(d_b\omega) - 3c_5\omega\sin(d_b\omega) + 6c_6 d_b\omega^2\Big) \end{aligned} \tag{198}$$

$$z(d_b) = \frac{2a_z + c_2\omega\cos(d_b\omega) + c_3\omega\sin(d_b\omega)}{2\omega^2} \tag{199}$$

$$\dot{x}(d_b) = \frac{8a_y + 8a_y\cos(d_b\omega) - 4a_x\sin(d_b\omega) + 3c_4\omega\cos(d_b\omega) + 3c_5\omega\sin(d_b\omega)}{4\omega} \tag{200}$$

$$\dot{y}(d_b) = -\frac{1}{2\omega}\Bigg(4a_x + 6c_6\omega + 4a_x\cos(d_b\omega) + 8a_y\sin(d_b\omega) + 6a_y d_b\omega - 3c_5\omega\cos(d_b\omega) + 3c_4\omega\sin(d_b\omega)\Bigg) \tag{201}$$

$$\dot{z}(d_b) = \frac{c_3\cos(d_b\omega) - c_2\sin(d_b\omega)}{2}, \tag{202}$$

where $d_b$ is the duration of the burn. The coefficients $c_1$-$c_6$ can be determined by setting $d_b = 0$ and plugging in the initial conditions:

$$\begin{aligned} c_1 &= \tfrac{2\dot{x}_0 - \omega y_0}{3\omega} & c_2 &= \tfrac{2z_0\omega^2 - 2a_z}{\omega} \\ c_3 &= 2\dot{z}_0 & c_4 &= \tfrac{4\omega\dot{x}_0 - 16a_y}{3\omega} \\ c_5 &= \tfrac{1}{6\omega}\left(24\omega^2 x_0 - 48a_x + 16\omega\dot{y}_0 + 64a_x\right) & c_6 &= \tfrac{1}{\omega}\left(2\omega^2 x_0 - 4a_x + \omega\dot{y}_0 + 4a_x\right) \end{aligned} \tag{203}$$

Thus, given the duration of the burn and the constant direction of the burn in three-dimensions, Equations 197–202 can be used for each burn in the sequence, and any coast phase can be propagated by the HCW STM, Equation 51. Mass loss is considered negligible during each burn in the sequence, but $a_0$ is adjusted for a second burn to account for mass lost during the first burn by the following equation,

$$a_{0_2} = \frac{a_0}{1 - d_{b1}\frac{a_0}{c}}, \tag{204}$$

where $d_{b1}$ is the duration of the first burn and $c$ is the (relatively high) constant effective exhaust velocity.

### 4.2.4 Targeted Teardrop.

This subsection describes how the desired teardrop is targeted for use in an optimization problem. Depending on mission requirements, a minimum-time or minimum-fuel solution may be required. Also, it may be required to enter the teardrop

directly, or simply enter the trajectory which after some time ends up forming the desired teardrop. The desired teardrop geometry, placement along the in-track direction, and out-of-plane motion must be prescribed. As explained in Chapter II, only two independent parameters need to be chosen to fully define the teardrop geometry. To ensure the inspector satellite maintains a minimum distance from the RSO, $D$ (the distance to the RSO at the closest approach) is chosen along with the period of the teardrop, $T_p$, as the two independent parameters. This determines the teardrop geometry and the two LROEs $a_e$ and $x_d$ through Equations 85 and 86.

The mission planner then defines the placement of the teardrop axis of symmetry, denoted as $y_T$, which is the $y$ location of both the teardrop cusp and the point of closest approach. According to Equation 76, it can be seen that the farthest radial position from the RSO is when $\beta = 0$. The point of closest approach in the radial direction is when $\beta = \pi$, and the radial position again reaches its farthest distance at $\beta = 2\pi$, as seen in Figure 15. Thus, with $y_T$ defining the teardrop axis of symmetry, it can be seen that $y = y_T$ when $\beta = \pi$, which means via Equation 78 that $y = y_T = y_d$ at that point, and happens halfway through the period, $P$, of the RSO. Thus, given $y_T$, $y_{d_0}$ can be calculated via Equation 71 as:

$$y_{d_0} = y_T + \frac{3}{2}\omega x_{d_0} t, \tag{205}$$

where $x_{d_0} = x_d$ is a constant and $t = \frac{1}{2}P$. Thus,

$$y_{d_0} = y_T + \frac{3}{4}\omega x_d P, \tag{206}$$

where this value for $y_{d_0}$ ensures the teardrop axis of symmetry is at the desired in-track location. The maximum out-of-plane distance of the teardrop can be chosen to coincide with the point of closest approach in the orbital plane as $Z_{top}$, where

$|Z_{top}| = z_{max}$. This means that the constant phase difference $\gamma$ between the in-plane and out-of-plane motion is $-90^o$ for a positive $Z_{top}$ and $90^o$ for a negative $Z_{top}$. Given this information, the set of states which will produce the desired teardrop can now be expressed as a function of $\beta$, the in-plane phase angle only, as,

$$x = \frac{-a_e}{2}\cos\beta + x_d \tag{207}$$

$$\dot{x} = \frac{a_e}{2}\omega\sin\beta \tag{208}$$

$$y = a_e\sin\beta + y_d = a_e\sin\beta + y_{d_0} - \frac{3}{2}x_d\beta \tag{209}$$

$$\dot{y} = a_e\omega\cos\beta - \frac{3}{2}\omega x_d \tag{210}$$

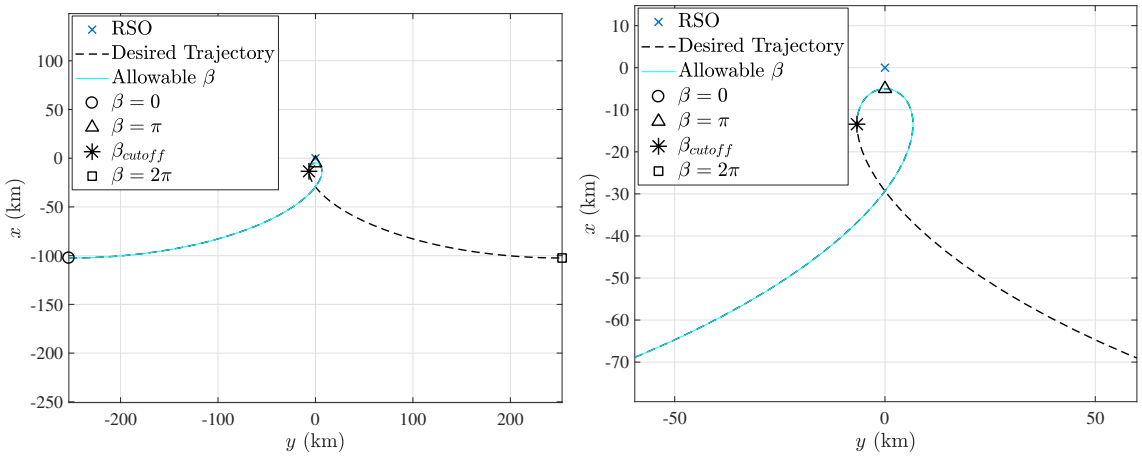
$$z = -Z_{top}\cos(\beta) \tag{211}$$

$$\dot{z} = Z_{top}\omega\sin(\beta). \tag{212}$$

Instead of also prescribing the value of $\beta$, it will be allowed to vary as a parameter to be optimized, which will allow the injection point into the teardrop trajectory to move with $\beta$ in order to further reduce the performance index. This injection angle is thus designed to have bounds, enabling the user to decide what range along the teardrop trajectory the inspector may enter. For this subproblem, the inspector satellite is prevented from entering the teardrop trajectory anywhere after $\beta_{cutoff}$, which has been chosen as the point of greatest width on the left side of the teardrop as shown in Figure 15. This ensures the inspector satellite enters the teardrop and has sufficient time to prepare for a potential burn as it approaches the cusp to repeat the teardrop trajectory. $\beta_{\text{cutoff}}$ is calculated by recognizing that $\dot{y} = 0$ at that point, and thus via Equation 79:

$$\beta_{cutoff} = 2\pi - \cos^{-1}\left(\frac{3x_d}{2a_e}\right), \tag{213}$$

where the calculated angle is ensured to be greater than $\pi$.




(a) Entire Teardrop Trajectory (b) Zoomed In

**Figure 15. Teardrop Example with Allowable Injection Range for $\beta$**

### 4.2.5 Optimization Problem Formulations.

The objective of this subproblem is to find the minimum-time and minimum-fuel solutions to inject an inspector satellite into and maintain a prescribed teardrop trajectory. The problems can be formulated such that a static optimization problem formulation is adequate to solve them, without discretizing them or using pseudospectral methods, at least for the initial guess and mid-fidelity solutions. This is done by using the analytic expressions for the propagation of a constant magnitude, constant direction burn for the HCW equations in the relative frame. The initial guess is first formulated and found via MATLAB's PSO, and then the mid-fidelity solution, using the PSO solution as an initial guess, is found via MATLAB's *fmincon*. Finally, the initial guess is used in GPOPS-II where a high-fidelity model is used to take into account continuous mass loss during each of the burns.

#### 4.2.5.1 Parameter Optimization Problem for PSO and NLP Solver.

This subsection outlines how the problem can be posed as a parameter optimization problem to be solved by both a PSO and an NLP solver to find both minimum-time and minimum-fuel solutions.

For the minimum-time problem, it is assumed that the typical solution for a minimum-time problem applies here [86]. That is, that the optimal control is to first use maximum acceleration to guide the inspector towards the terminal conditions and then second, to apply the same maximum acceleration (typically approximately in the reverse direction) to meet the terminal constraints. The control is thus parameterized into two segments, namely a burn-burn sequence, where the direction of the thrust is assumed to be constant in the relative frame during each segment. Thus, the control is parameterized into $\alpha_1$, $\phi_1$, $\alpha_2$, and $\phi_2$ where the subscripts represent burns one and two. These parameters, along with the final time, $t_f$, and the fraction of the final time at which to switch to burn two, $t_{2_f}$, are the fewest variables needed to complete the parameterization of the control. These variables and $\beta$ are the parameters to be optimized,

$$\chi = [\alpha_1, \alpha_2, \phi_1, \phi_2, t_{2_f}, t_f, \beta], \tag{214}$$

where the upper and lower bounds are:

$$\chi_l = \left[0, 0, -\frac{\pi}{2}, -\frac{\pi}{2}, 0, 0, 0\right] \tag{215}$$

$$\chi_u = \left[2\pi, 2\pi, \frac{\pi}{2}, \frac{\pi}{2}, 1, t_{max}, \beta_{cutoff}\right]. \tag{216}$$

For the minimum-time problem, the performance index is the final time, $t_f$,

$$J = t_f, \tag{217}$$

subject to the equality constraints (which are terminal constraints),

$$h = \begin{bmatrix} x_f + \frac{a_e}{2}\cos\beta - x_d \\ \dot{x}_f - \frac{a_e}{2}\omega\sin\beta \\ y_f - a_e\sin\beta - y_{d_0} + \frac{3}{2}x_d\beta \\ \dot{y}_f - a_e\omega\cos\beta + \frac{3}{2}\omega x_d \\ z_f + Z_{top}\cos(\beta) \\ \dot{z}_f - Z_{top}\omega\sin(\beta) \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{bmatrix} = 0, \tag{218}$$

where the final state after the two burns, $X_f = [x_f, y_f, z_f, \dot{x}_f, \dot{y}_f, \dot{z}_f]^T$, minus the solution to Equations 207–212 must be equal to zero. When utilizing the NLP solver, these are treated as equality constraints, whereas when using the PSO, they are appended to the cost function. Thus the actual cost function the PSO attempts to minimize in order to also meet terminal constraints is,

$$\tilde{J} = J + \sum_{r=1}^{6} W_r h_r^2(\chi), \tag{219}$$

where $W_r$ must be chosen to scale the positions and velocities appropriately. Given the seven optimization parameters in $\chi$, the burn-burn sequence can be analytically propagated by using Equations 197–202 twice, where $d_{b1} = t_{2_f} t_f$ and $d_{b2} = t_f(1 - t_{2_f})$, and the equality constraints can be evaluated.

For the minimum-fuel problem there are several changes. First, a coasting phase is introduced between the two burns and second, the final time, $t_f$, for the burn-coast-burn sequence is now fixed, where the optimizer seeks to maximize the coast time, or

minimize the total engine-on time. The parameters to be optimized thus become

$$\chi = [\alpha_1, \alpha_2, \phi_1, \phi_2, t_{c_f}, t_{2_f}, \beta] \tag{220}$$

where $t_{2_f}$ is the fraction of the fixed final time when burn two starts and produces the time at which burn two starts, $t_{2_0} = t_{2_f} t_f$. Then, $t_{c_f}$ is the fraction of $t_{2_0}$ when coasting starts (or the first burn ends) and produces the time at which coasting starts, $t_{c_0} = t_{c_f} t_{2_0}$. These first six variables in $\chi$ represent the minimum set needed to parameterize the control for the burn-coast-burn sequence. Thus, the propagation for the two burns works the same as before, but after the first burn the states are propagated analytically with the HCW STM by the total coast time, $d_c = t_{2_0} - t_{c_0}$, and the final conditions of the coast phase become the initial conditions for the second burn. The lower and upper bounds are

$$\chi_l = \left[0, 0, -\frac{\pi}{2}, -\frac{\pi}{2}, 0, 0, 0\right] \tag{221}$$

$$\chi_u = \left[2\pi, 2\pi, \frac{\pi}{2}, \frac{\pi}{2}, 1, 1, \beta_{cutoff}\right]. \tag{222}$$

The cost for the NLP solver and appended cost function for the PSO are then,

$$J = t_f - t_{2_0} + t_{c_0} \tag{223}$$

$$\tilde{J} = J + \sum_{r=1}^{6} W_r h_r^2(\chi), \tag{224}$$

where the equality constraints are the same as in Equation 218, except that $X_f$ is now the final state after the burn-coast-burn sequence.

Thus, for both the minimum-time and minimum-fuel problem formulations, there are (rather long) analytic expressions which generate $X_f = [x_f, y_f, z_f, \dot{x}_f, \dot{y}_f, \dot{z}_f]$ and thus the equations for the equality constraints (terminal constraints) are a function of

seven optimization parameters. The minimum-time and minimum-fuel formulations will first be solved by MATLAB's PSO, and then also by MATLAB's *fmincon*, using the PSO solution as an initial guess for the NLP solver. Additionally, the Radau pseudospectral method will be employed via GPOPS-II to attempt to improve the accuracy of the solution, incorporating time-varying mass loss instead of only updating the mass after each burn.

#### 4.2.5.2 GPOPS-II Problem Formulation.

When utilizing GPOPS-II, the problem is split into two separate multi-phase optimization problems, where the second optimization problem solves for the optimal control to repeat the teardrop trajectory once the inspector reaches the cusp, or intersection point.

The first optimization problem is split up into two phases for the minimum-time problem, where the first phase is a constant direction burn but where mass loss is accounted for at each discretized point during the burn, as is the case for all the burns in this pseudospectral formulation. Phase two is another constant direction burn, with the same terminal constraints as shown in Equation 218. The two phases allow for discontinuities in the control, but the states and times are linked from phase one to phase two. The minimum-fuel problem has an additional coast phase in between the two burn phases. Therefore the first burn is phase one, the coast is phase two, and the second burn is phase three for the minimum-fuel problem.

The next optimization problem starts once the natural motion from the terminal conditions of the first optimization problem have been propagated to $\beta_{cutoff}$. The first phase is the natural motion from $\beta_{cutoff}$ towards the cusp, to allow the solution to exit this phase when required in order to make the optimal finite burn to repeat the teardrop pattern, which is the second phase of this second optimization problem.

The terminal constraints for the second phase are the same as in Equation 218, but the bounds on $\beta$ change the allowable entry for $\beta$ to be only the right hand portion of the teardrop in order to generate a finite burn to repeat the teardrop pattern. Thus, the bounds on $\beta$ are $\frac{P-T_p}{2}\omega \leq \beta \leq \pi$.

The cost functions for the first optimization problem are:

$$J = t_f^{(2)} \tag{225}$$

$$J = t_f^{(3)} - t_f^{(2)} + t_f^{(1)}, \tag{226}$$

for the minimum-time and minimum-fuel formulations respectively, where the superscripts denote the phases for each formulation, and the final times of each phase are cumulative, meaning that for example $t_f^{(3)}$ is the elapsed time of all three phases combined for the minimum-fuel problem formulation. The cost function for the second optimization problem is the same for both the minimum-time and minimum-fuel formulations,

$$J = t_f^{(2)} - t_f^{(1)}, \tag{227}$$

since the minimum-time and minimum-fuel problems have identical solutions in this case.

### 4.2.6 Simulations and Results.

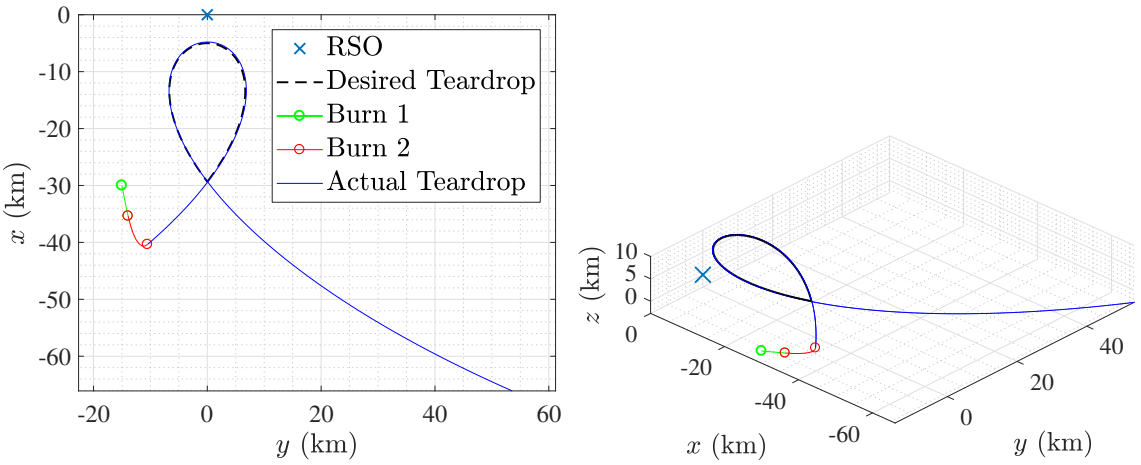
All simulations take place near an RSO in GEO with the parameters shown in Table 4.

To begin, a typical resulting trajectory when using the PSO is shown for a minimum-time formulation in Figure 16 in both two and three dimensions. The cost for this solution is 27.27 minutes to inject. A typical solution for a minimum-

**Table 4. Problem B-1 Simulation Parameters**

| Spacecraft Properties | Teardrop Parameters | Initial Conditions |
|---|---|---|
| $a_0 = 0.02$ N/kg<br>$c = 3.33$ km/s | $D = -5$ km<br>$T_p = P/3$<br>$y_T = 0$ km<br>$Z_{top} = 10$ km | $x = -30$ km $\quad \dot{x} = 0$ km/s<br>$y = -15$ km $\quad \dot{y} = 0$ km/s<br>$z = 0$ km $\quad \dot{z} = 0$ km/s |

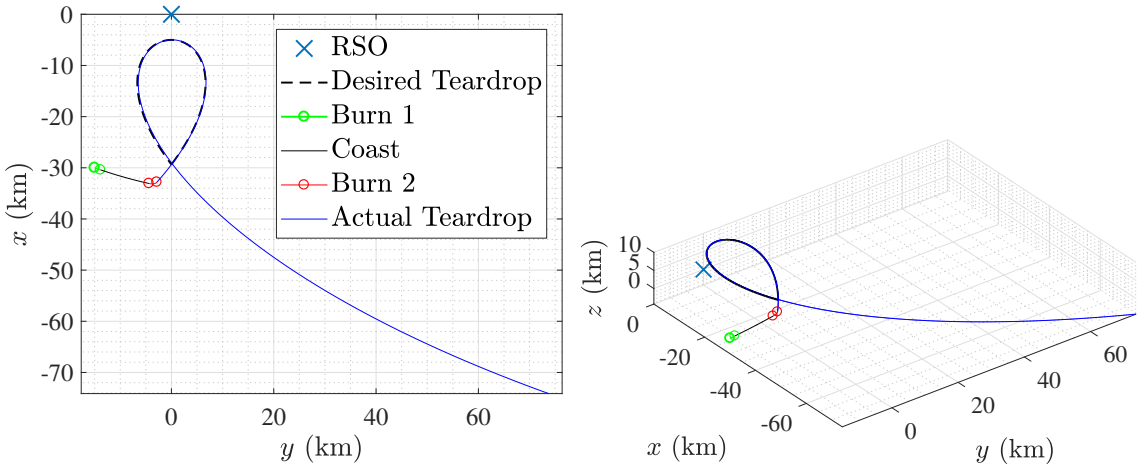
fuel formulation with a fixed final time of 35 minutes is shown in Figure 17 and has an engine-on time of 11.27 minutes. Thus, comparing these minimum-time and minimum-fuel solutions, the required engine-on time decreases dramatically if the minimum-time case is allowed 7-8 more minutes and the fuel usage is minimized, as would be expected.




(a) 2-D (b) 3-D

**Figure 16. Problem B-1, PSO Min Time Solution**

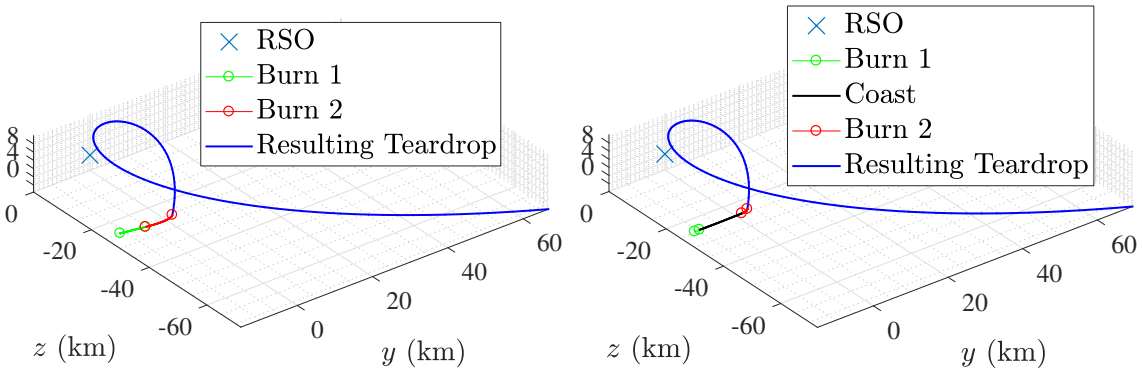
Now the parameter optimization problem is solved by using an NLP solver, namely MATLAB's *fmincon* solver. The NLP solver must be provided an initial guess. If no good initial guess exists and an arbitrary (random) guess is provided, then the NLP solver may not converge to a feasible solution. However, if an initial guess is provided with the PSO solution, the NLP algorithm quickly converges with a computation time of about one second. A minimum-time solution obtained with the NLP solver given an initial guess from the PSO is shown in Figure 18 (a). The computation time




(a) 2-D (b) 3-D

**Figure 17. Problem B-1, PSO Min Fuel Solution, $t_f = 35$ minutes**

for this particular solution is 0.8 seconds with a final time of 24.94 minutes, which is a little less than the PSO solution. Figure 18 (b) shows a minimum-fuel solution, again by using a PSO solution as an initial guess and then refining the result with the NLP solver. The computation time for this scenario is again on the order of one second with a total engine-on time of 11.00 minutes, once again refining the PSO solution.

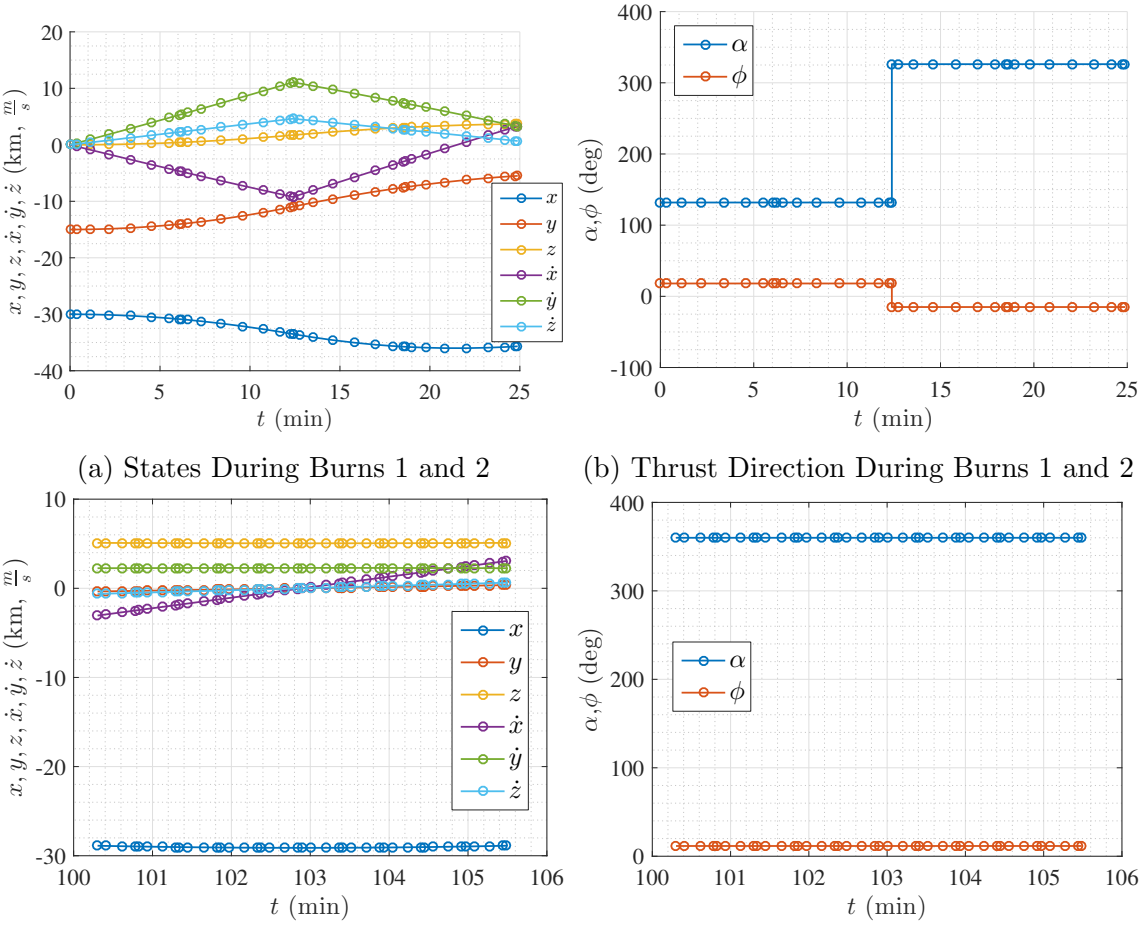



(a) Min Time Solution (b) Min Fuel Solution, $t_f = 35$ minutes

**Figure 18. Problem B-1, *fmincon* Min Time & Fuel Solutions - Initial Guess with PSO**

GPOPS-II is now used as a pseudospectral solver to further refine the results, allow for time-varying mass loss, and find the optimal finite burn to repeat the teardrop motion. The same parameters are used as in Table 4, and both a minimum-time

and minimum-fuel solution is presented, again using an initial guess from the PSO for each formulation. For the minimum-time solution, Figure 19 shows the optimal states during burns one and two in (a) and also the optimal direction of the thrust during burns one and two in (b), where the circle data points represent the collocation points used by GPOPS-II. Figure 19 (c) and (d) show the optimal states and thrust direction for the finite repeat burn, showing how just one constant direction burn is required to repeat the pattern since they are intersecting trajectories, and the minimum time to repeat the teardrop is the same as the minimum fuel to repeat.




(a) States During Burns 1 and 2

(b) Thrust Direction During Burns 1 and 2

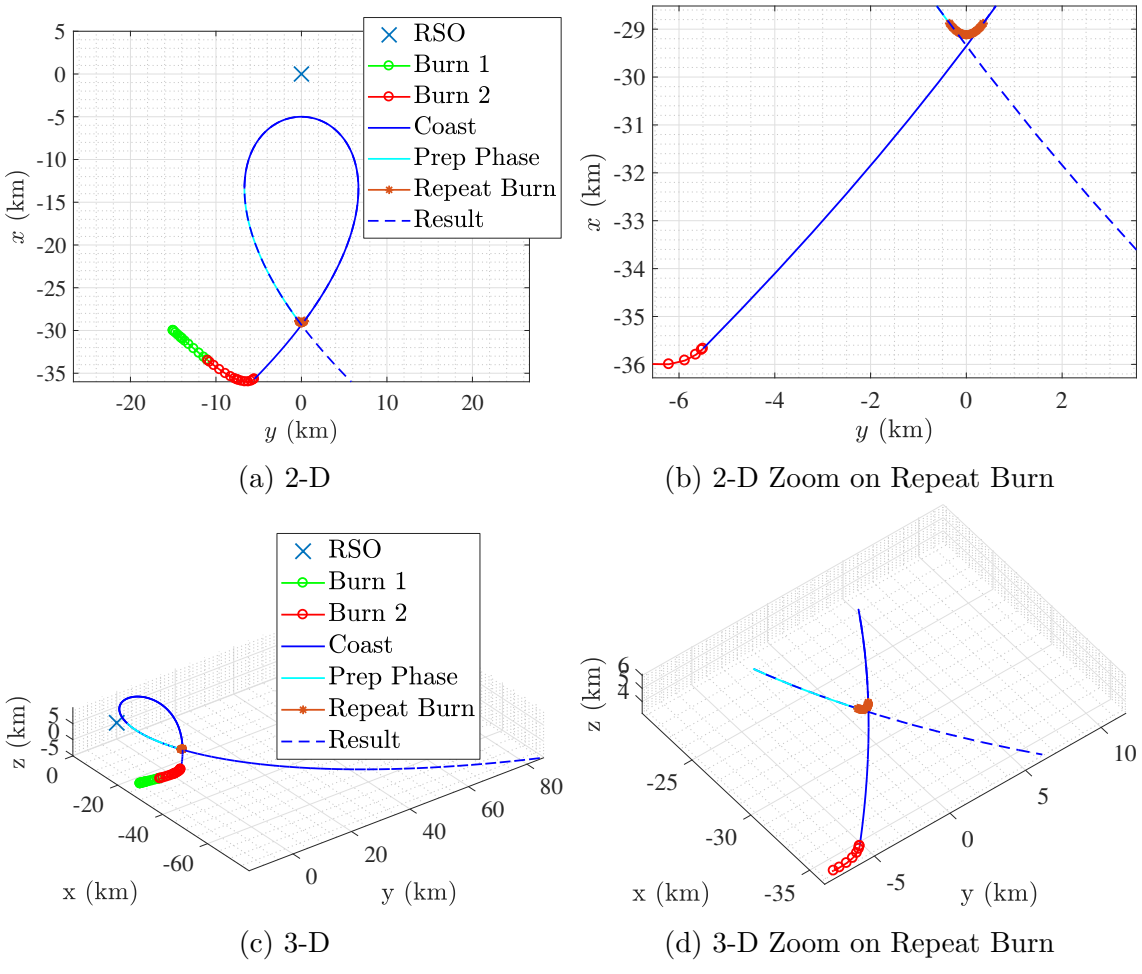
(c) States During Repeat Burn

(d) Thrust Direction During Repeat Burn

**Figure 19. Problem B-1, GPOPS-II Min Time Solution - Initial Guess with PSO**

Figure 20 shows the resulting trajectories from the minimum-time solution, where

the results from both optimization problems for the injection and repeat problems are shown. That is, burns one and two in the figures correspond to phases one and two of the first optimization problem. The coasting portion in the figures is the resulting motion from the first optimization problem. Then, the preparation phase in the figures for the repeat burn is the first phase of the second optimization problem, and the repeat burn is the second phase of the second optimization problem, maneuvering the satellite back into the teardrop. The initial guess again comes from the PSO, which GPOPS-II refines and also accounts for mass loss during the two burns, producing a solution of 24.88 minutes to inject.




(a) 2-D

(b) 2-D Zoom on Repeat Burn

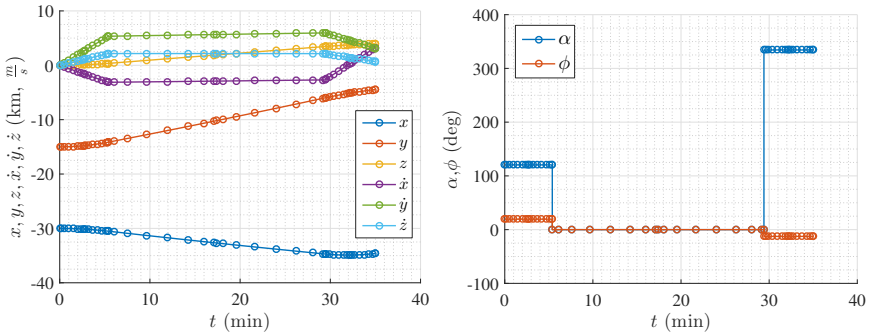
(c) 3-D

(d) 3-D Zoom on Repeat Burn

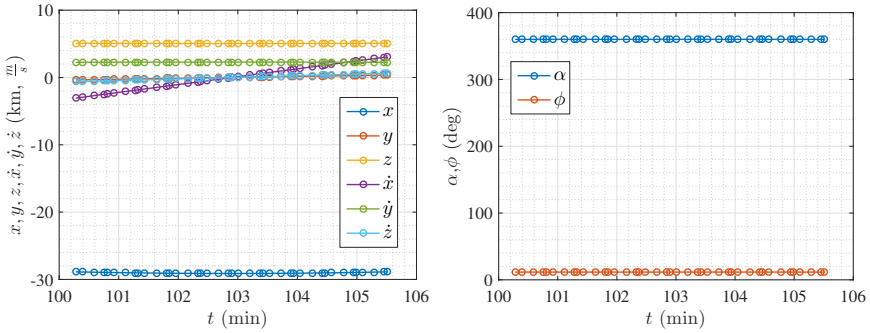
**Figure 20. Problem B-1, GPOPS-II Min Time Trajectories, Inject and Repeat Burns**

For the minimum-fuel solution, Figure 21 shows the optimal states during burn

one, the coasting phase, and burn two in (a) and also the optimal direction of the thrust during burns one and two in (b), where the angles during the coast are simply set to zero since they don't have any influence during the coasting phase. Figure 21 (c) and (d) show the optimal states and thrust direction for the repeat burn, and look approximately identical to the minimum-time solution, as expected.




(a) States for Burn-Coast-Burn (b) Thrust Direction, Burns 1-2

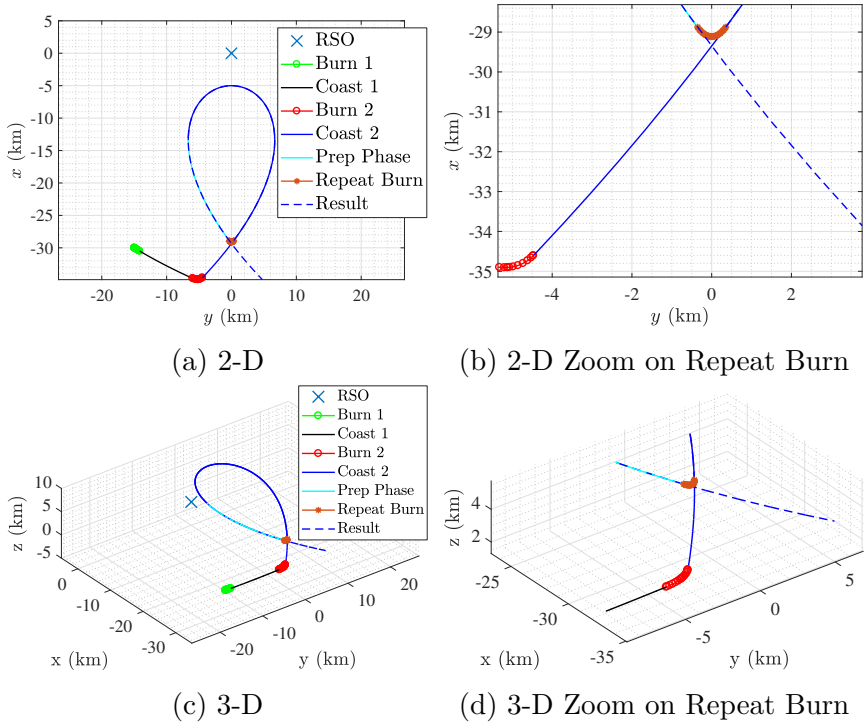



(c) States for Repeat Burn (d) Thrust Direction, Repeat

**Figure 21. Problem B-1, GPOPS-II Min Fuel Solution, $t_f = 35$ minutes**

Figure 22 shows the resulting trajectories, where the results from both optimization problems composing the minimum-fuel solution for injection into and maintenance of the teardrop trajectory are shown. That is, burns one and two correspond to phases one and three of the first optimization problem, while the coast between burns is phase two. The resulting motion from the first optimization problem is then shown until the preparation phase for the repeat burn which is the first phase of the second optimization problem. The repeat burn is the second phase of the second optimization problem, maneuvering the satellite back into the teardrop. The solution

from the PSO is again used as the initial guess, which the pseudospectral algorithm refines, accounting for mass loss during the two burns and satisfying constraints to a defined tolerance, producing a solution with an engine-on time of 10.99 minutes, further reducing the overall cost.




(a) 2-D (b) 2-D Zoom on Repeat Burn

(c) 3-D (d) 3-D Zoom on Repeat Burn

**Figure 22. B-1, GPOPS-II Min Fuel Trajectories, $t_f = 35$ minutes, Inject, Repeat Burns**

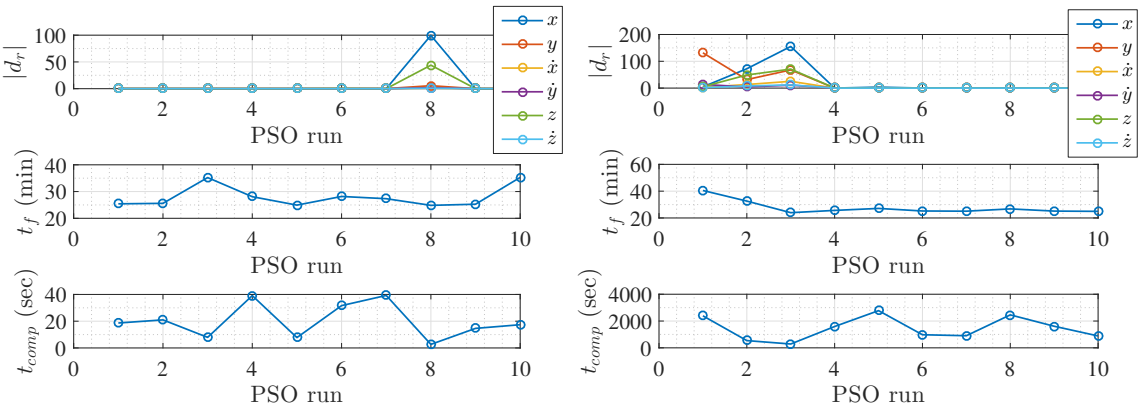
A summary of the simulation results can be seen in Table 5, where all CPU times provided throughout this research were obtained using a standard laptop with eight 2.4 GHz CPUs and 16 GB RAM. When using MATLAB's PSO, default settings were used with the exception of setting the swarm size to 350–500 particles or slightly increasing the maximum number of iterations. When using MATLAB's *fmincon*, default settings were used with the exception of increasing the maximum number of iterations and function evaluations. When using GPOPS-II, IPOPT was used as the NLP solver, the default GPOPS-II tolerances were used, and no special treatment regarding the settings were required to obtain the solutions.

**Table 5. Problem B-1 Simulation Results**

| | **Min Time** | | **Min Fuel** ($t_f = 35$ min) | |
|---|---|---|---|---|
| | J (min) | CPU Time (s) | J (min) | CPU Time (s) |
| PSO (typical run) | 27.27 | ≈20 | 11.27 | ≈20 |
| *fmincon* | 24.94 | ≈1 | 11.00 | ≈1 |
| GPOPS-II | 24.88 | ≈1 | 10.99 | ≈1 |

### 4.2.7 PSO Performance.

An analysis of the performance of the PSO algorithm for the minimum-time formulation is presented, comparing the analytic propagation of the two burns vs. numerically propagating them via a typical numerical propagator (MATLAB's *ode45*). Figure 23 shows the constraint violations (denoted $|d_r|$ here), costs ($t_f$), and CPU times ($t_{comp}$) for 10 PSO runs with 400 particles each for both the analytic propagation and numerical propagation methods. These results highlight several things. First, the PSO finds a good solution 70-90% of the time with the parameters shown in Table 4, varying in performance due to its stochastic nature, and is not yet reliable enough for a user to be confident in obtaining a standalone solution every time it's run. However, it may be an effective way to obtain an initial guess, since when using the analytic method it takes approximately 20 seconds on average to find a solution, whereas the numerical method takes about 24 minutes (72 times longer) on average to obtain a solution. Thus, using the developed expressions to analytically propagate the burns provides a relatively quick way to use a metaheuristic algorithm to obtain an initial guess, and may afford the use of more particles in the PSO, further improving the solution.




(a) Analytic Propagation (b) Numerical Propagation

**Figure 23. PSO Performance, Analytic & Numerical Propagation of Two Burns**

### 4.2.8 Problem B-1 Conclusion.

Multiple optimization methods have been successfully employed to find both minimum-time and minimum-fuel solutions for an inspector satellite to inject into and maintain a prescribed three-dimensional teardrop trajectory. An analytic expression for the six HCW states after a constant acceleration magnitude and constant direction burn can be successfully developed and utilized to propagate burns in defined sequences which significantly decreases the solution time of the PSO (by a factor of 72 for the case shown) compared to using a numerical propagator. The PSO generates good initial guesses for MATLAB's *fmincon* and GPOPS-II, helping those methods avoid convergence issues. The analytic propagation of the two burns also enables a fast computation time for the NLP solver, allowing it to quickly converge to a local minimum given the PSO solution as an initial guess. The formulation using GPOPS-II successfully accounts for mass loss due to thrusting at each discretized point, and further refines the solutions obtained from the PSO and NLP solver. The LROEs and teardrop parameters can be successfully used to constrain terminal conditions to the desired teardrop trajectory, where the user only needs to pick two teardrop parameters along with $y_T$ and $Z_{top}$ to define the three-dimensional region in which the

quasi-hovering is desired. The bounds on the entry angle, $\beta$, enable the user to specify where the inspector is allowed to enter, and the entire trajectory is parameterized by this one angle. Thus, the algorithms developed here produce optimal guidance for on/off, finite-thrust engines, where negligible mass loss assumptions pertaining to the PSO and NLP solutions are more appropriate for low-thrust engines, and GPOPS-II can be used to improve accuracy for higher-thrust engines.

## 4.3 Problem B-2

### 4.3.1 Overview.

Problem B-2 uses a burn-coast-burn sequence like Problem B-1 did for the fuel-optimal maneuvers, but focuses on maneuvering into an NMC, subject to various lighting constraints. The entire analytic expression for the propagation of the burn-coast-burn sequence is presented, which is developed for the purpose of finding analytic derivatives so that they can be used in the NLP solvers. For the first part of this subproblem, the sunlight constraints are developed, as well as new initial guess methods in order to improve upon the initial guess used in Problem B-1, which was the solution from MATLAB's PSO. Simulations and results are then presented with the sunlight constraints enforced, and an analysis of the initial guess methods coupled with varying levels of user-supplied derivatives provided to the NLP solvers is presented. The last part of this subproblem will also develop field-of-view constraints in order to keep the Earth and the Moon outside of the field of view in the desired natural motion, and simulations and results will be presented for cases where these additional constraints are enforced in addition to the sunlight constraints.

### 4.3.2 Equations of Motion and Control Definition.

For the initial guesses and the mid-fidelity model, Problem B-2 uses the HCW equations of motion where again for each burn the acceleration terms are constants, as in Equations 194–196. For the high-fidelity model, Equations 4–6 are used, which are the CNERMs.

### 4.3.3 Analytic Propagation of a Burn-Coast-Burn Sequence.

For all scenarios in this subproblem, minimum-fuel solutions are sought while using a burn-coast-burn sequence. When using the HCW equations of motion and

the control defined previously, the six final states after the burn-coast-burn sequence have an analytic expression. Using MATLAB's *dsolve* and Symbolic Toolbox, the final states, $X_f = [x_f, y_f, z_f, \dot{x}_f, \dot{y}_f, \dot{z}_f]$, are:

$$
\begin{aligned}
x_f = \frac{-1}{\omega^2\left(c - a_0 t_{2_f} t_{c_f} t_f\right)}\Bigg(
&2c\omega\dot{y}_0\cos\left(t_f\omega\right) - 2c\omega\dot{y}_0 - 4c\omega^2 x_0 - c\omega\dot{x}_0\sin\left(t_f\omega\right) \\
&- a_0 c\cos\left(\alpha_2\right)\cos\left(\phi_2\right) + 3c\omega^2 x_0\cos\left(t_f\omega\right) + a_0 c\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
&+ 2a_0 c\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
&- a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) + 2a_0 t_{2_f} t_{c_f} t_f\omega\dot{y}_0 \\
&- 2a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
&+ 2a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
&- a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
&+ 4a_0 t_{2_f} t_{c_f} t_f\omega^2 x_0 + a_0 c\cos\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
&+ 2a_0 c\cos\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
&- 2a_0 c\sin\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
&+ a_0 c\sin\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) - 2a_0 c t_f\omega\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
&- 3a_0 t_{2_f} t_{c_f} t_f\omega^2 x_0\cos\left(t_f\omega\right) + 2a_0 c t_{2_f} t_f\omega\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
&+ 2{a_0}^2 {t_{2_f}}^2 {t_{c_f}}^2 {t_f}^2\omega\cos\left(\phi_1\right)\sin\left(\alpha_1\right) - 2a_0 t_{2_f} t_{c_f} t_f\omega\dot{y}_0\cos\left(t_f\omega\right) \\
&- {a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) + a_0 t_{2_f} t_{c_f} t_f\omega\dot{x}_0\sin\left(t_f\omega\right) \\
&- 2{a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
&+ {a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
&+ 2{a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
&- 2{a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
&+ {a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
&- 2a_0 c t_{2_f} t_{c_f} t_f\omega\cos\left(\phi_1\right)\sin\left(\alpha_1\right)\Bigg)
\end{aligned}
\tag{228}
$$

$$
\begin{aligned}
y_f = \frac{-1}{2\omega^2\left(c - a_0 t_{2_f} t_{c_f} t_f\right)}\Bigg( & 4c\omega\dot{x}_0 - 2c\omega^2 y_0 - 4c\omega\dot{x}_0\cos\left(t_f\omega\right) - 8c\omega\dot{y}_0\sin\left(t_f\omega\right) \\
& + 12ct_f\omega^3 x_0 + 6ct_f\omega^2\dot{y}_0 - 8a_0c\cos\left(\phi_2\right)\sin\left(\alpha_2\right) - 12c\omega^2 x_0\sin\left(t_f\omega\right) \\
& + 8a_0c\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) - 4a_0c\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - 4a_0 t_{2_f} t_{c_f} t_f\omega\dot{x}_0 - 8a_0c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 4a_0c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - 4a_0c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - 8a_0c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) + 2a_0 t_{2_f} t_{c_f} t_f\omega^2 y_0 \\
& + 3a_0ct_f{}^2\omega^2\cos\left(\phi_2\right)\sin\left(\alpha_2\right) + 8a_0c\cos\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& - 4a_0c\cos\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& + 4a_0c\sin\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& + 4a_0ct_f\omega\cos\left(\alpha_2\right)\cos\left(\phi_2\right) + 8a_0c\sin\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& - 12a_0 t_{2_f} t_{c_f} t_f{}^2\omega^3 x_0 - 6a_0 t_{2_f} t_{c_f} t_f{}^2\omega^2\dot{y}_0 - 6a_0{}^2 t_{2_f}{}^2 t_{c_f}{}^2 t_f{}^3\omega^2\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 3a_0{}^2 t_{2_f}{}^3 t_{c_f}{}^3 t_f{}^3\omega^2\cos\left(\phi_1\right)\sin\left(\alpha_1\right) - 4a_0ct_{2_f} t_f\omega\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& + 12a_0 t_{2_f} t_{c_f} t_f\omega^2 x_0\sin\left(t_f\omega\right) + 3a_0ct_{2_f}{}^2 t_f{}^2\omega^2\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& - 4a_0{}^2 t_{2_f}{}^2 t_{c_f}{}^2 t_f{}^2\omega\cos\left(\alpha_1\right)\cos\left(\phi_1\right) + 4a_0 t_{2_f} t_{c_f} t_f\omega\dot{x}_0\cos\left(t_f\omega\right) \\
& + 8a_0 t_{2_f} t_{c_f} t_f\omega\dot{y}_0\sin\left(t_f\omega\right) - 8a_0{}^2 t_{2_f} t_{c_f} t_f\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 4a_0{}^2 t_{2_f} t_{c_f} t_f\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) - 6a_0ct_{2_f} t_f{}^2\omega^2\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& - 3a_0ct_{2_f}{}^2 t_{c_f}{}^2 t_f{}^2\omega^2\cos\left(\phi_1\right)\sin\left(\alpha_1\right) + 6a_0ct_{2_f} t_{c_f} t_f{}^2\omega^2\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 8a_0{}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& - 4a_0{}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + 4a_0{}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + 8a_0{}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 4a_0ct_{2_f} t_{c_f} t_f\omega\cos\left(\alpha_1\right)\cos\left(\phi_1\right)\Bigg)
\end{aligned}
\tag{229}
$$

$$
\begin{aligned}
z_f = \frac{-1}{\omega^2\left(c - a_0 t_{2_f} t_{c_f} t_f\right)}\Bigg( & a_0 c\cos\left(t_f\omega\right)\sin\left(\phi_1\right) - c\omega\dot{z}_0\sin\left(t_f\omega\right) - a_0 c\sin\left(\phi_2\right) \\
& - c\omega^2 z_0\cos\left(t_f\omega\right) - a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& - a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\sin\left(\phi_1\right) + a_0 c\cos\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\sin\left(\phi_2\right) \\
& + a_0 c\sin\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\sin\left(\phi_2\right) - {a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& + a_0 t_{2_f} t_{c_f} t_f\omega^2 z_0\cos\left(t_f\omega\right) + {a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& + {a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\sin\left(\phi_1\right) + a_0 t_{2_f} t_{c_f} t_f\omega\dot{z}_0\sin\left(t_f\omega\right)\Bigg)
\end{aligned}
\tag{230}
$$

$$
\begin{aligned}
\dot{x}_f = \frac{-1}{\omega\left(c - a_0 t_{2_f} t_{c_f} t_f\right)}\Bigg( & 2a_0 c\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) - 2c\omega\dot{y}_0\sin\left(t_f\omega\right) \\
& - 2a_0 c\cos\left(\phi_2\right)\sin\left(\alpha_2\right) - 3c\omega^2 x_0\sin\left(t_f\omega\right) - c\omega\dot{x}_0\cos\left(t_f\omega\right) \\
& - a_0 c\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - 2a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - 2a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 2a_0 c\cos\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& - a_0 c\cos\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& + a_0 c\sin\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& + 2a_0 c\sin\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& + 3a_0 t_{2_f} t_{c_f} t_f\omega^2 x_0\sin\left(t_f\omega\right) + a_0 t_{2_f} t_{c_f} t_f\omega\dot{x}_0\cos\left(t_f\omega\right) \\
& + 2a_0 t_{2_f} t_{c_f} t_f\omega\dot{y}_0\sin\left(t_f\omega\right) - 2{a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + {a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + 2{a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& - {a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + {a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + 2{a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right)\Bigg)
\end{aligned}
\tag{231}
$$

$$
\begin{aligned}
\dot{y}_f = \frac{1}{\omega\left(c - a_0 t_{2_f} t_{c_f} t_f\right)}\Bigg( & 4c\omega\dot{y}_0\cos\left(t_f\omega\right) - 3c\omega\dot{y}_0 - 6c\omega^2 x_0 - 2c\omega\dot{x}_0\sin\left(t_f\omega\right) \\
& - 2a_0 c\cos\left(\alpha_2\right)\cos\left(\phi_2\right) + 6c\omega^2 x_0\cos\left(t_f\omega\right) \\
& + 2a_0 c\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) + 4a_0 c\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& - 2a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) + 3a_0 t_{2_f} t_{c_f} t_f\omega\dot{y}_0 \\
& - 4a_0 c\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 4a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& - 2a_0 c\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) + 6a_0 t_{2_f} t_{c_f} t_f\omega^2 x_0 \\
& + 2a_0 c\cos\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& + 4a_0 c\cos\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& - 4a_0 c\sin\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_2\right)\sin\left(\alpha_2\right) \\
& + 2a_0 c\sin\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_2\right)\cos\left(\phi_2\right) \\
& - 3a_0 c t_f\omega\cos\left(\phi_2\right)\sin\left(\alpha_2\right) - 6a_0 t_{2_f} t_{c_f} t_f\omega^2 x_0\cos\left(t_f\omega\right) \\
& + 3a_0 c t_{2_f} t_f\omega\cos\left(\phi_2\right)\sin\left(\alpha_2\right) + 3{a_0}^2 {t_{2_f}}^2 {t_{c_f}}^2 {t_f}^2\omega\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& - 4a_0 t_{2_f} t_{c_f} t_f\omega\dot{y}_0\cos\left(t_f\omega\right) - 2{a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + 2a_0 t_{2_f} t_{c_f} t_f\omega\dot{x}_0\sin\left(t_f\omega\right) - 4{a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 2{a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& + 4{a_0}^2 t_{2_f} t_{c_f} t_f\cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& - 4{a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\cos\left(\phi_1\right)\sin\left(\alpha_1\right) \\
& + 2{a_0}^2 t_{2_f} t_{c_f} t_f\sin\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\cos\left(\alpha_1\right)\cos\left(\phi_1\right) \\
& - 3a_0 c t_{2_f} t_{c_f} t_f\omega\cos\left(\phi_1\right)\sin\left(\alpha_1\right)\Bigg)
\end{aligned}
\tag{232}
$$

$$
\begin{aligned}
\dot{z}_f = \frac{1}{\omega\left(c - a_0 t_{2_f} t_{c_f} t_f\right)}\Bigg( & c\omega\dot{z}_0 \cos\left(t_f\omega\right) + a_0 c \sin\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& - c\omega^2 z_0 \sin\left(t_f\omega\right) - a_0 c \cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& + a_0 c \sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\sin\left(\phi_1\right) + a_0 c \cos\left(t_{2_f} t_f\omega\right)\sin\left(t_f\omega\right)\sin\left(\phi_2\right) \\
& - a_0 c \sin\left(t_{2_f} t_f\omega\right)\cos\left(t_f\omega\right)\sin\left(\phi_2\right) - {a_0}^2 t_{2_f} t_{c_f} t_f \sin\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& + a_0 t_{2_f} t_{c_f} t_f \omega^2 z_0 \sin\left(t_f\omega\right) + {a_0}^2 t_{2_f} t_{c_f} t_f \cos\left(t_{2_f} t_{c_f} t_f\omega\right)\sin\left(t_f\omega\right)\sin\left(\phi_1\right) \\
& - {a_0}^2 t_{2_f} t_{c_f} t_f \sin\left(t_{2_f} t_{c_f} t_f\omega\right)\cos\left(t_f\omega\right)\sin\left(\phi_1\right) - a_0 t_{2_f} t_{c_f} t_f \omega \dot{z}_0 \cos\left(t_f\omega\right)\Bigg).
\end{aligned}
\tag{233}
$$

These equations are a function known parameters and control variables, where the control variables will form the main optimization variables. The known parameters are: the initial state, $X_0 = [x_0, y_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0]$, the initial acceleration magnitude, $a_0$, and the constant effective exhaust velocity, $c$, the mean motion, $\omega$, and the fixed final time of the burn-coast-burn sequence, $t_f$, which is given beforehand for a minimum-fuel maneuver. The control variables are: the in-plane and out-of-plane angles for each burn, $\alpha_1$, $\phi_1$, $\alpha_2$, and $\phi_2$, and $t_{2_f}$ and $t_{c_f}$, which along with $t_f$ are used to define the duration of each phase in the sequence as in Problem B-1 for the minimum-fuel formulation. Note that in Equations 228–233, the term $c - a_0 t_{2_f} t_{c_f} t_f$ appears in the denominator. This term results from the fact that these equations account for the mass lost after the first burn. For this term to approach zero, a highly unlikely and unusual combination of the effective exhaust velocity, $c$, the initial thrust-to-mass ratio, $a_0$, and duration of the first burn, $t_{2_f} t_{c_f} t_f$, would have to occur. Physically, it would mean that the thrust-to-mass ratio for the second burn is approaching infinity, which is not possible with a constant thrust since the fuel would run out before the mass of the satellite approached zero. Therefore, for real-world problems, this possibility can be ignored.

### 4.3.4 Sunlight Constraints for NMC.

Upon entering an NMC about an RSO, it is desirable to have favorable lighting conditions, meaning that it is desirable that the RSO is lit by the Sun from the point of view of the inspector satellite. Thus, this section develops two different sunlight constraints. The first constraint will be called a hard or tight sunlight constraint, where the inspector must enter the NMC between the RSO and the Sun at a point along the vector from the Sun to the RSO, or $r_{s2c}$, when projected onto the orbital plane of the RSO, (where the subscript $c$ represents the chief satellite, or RSO). One nice property about an NMC that encircles the RSO is that if the inspector satellite obtains favorable lighting upon entry into the NMC, then the projection of $r_{s2c}$ will approximately follow the inspector satellite around the NMC, at least for the course of several days. Thus, the inspector can enter the NMC approximately in phase with the Sun, or out of phase, or somewhere in between. The second sunlight constraint developed in this study will be called a soft or relaxed sunlight constraint, meaning that a margin is allowed from the hard constraint entry point by a prescribed angle in both directions from the entry point, denoted as $\theta_s$.

Given the definition of the two types of sunlight constraints, the vector from the Sun to the RSO, $r_{s2c}$, must be found and expressed in the LVLH frame of the RSO. To do this, $r_{s2c}$ is first found in the J2000 Earth-Centered-Inertial (ECI) frame, by using the algorithms supplied by Vallado [13]. First, given the total time allowed for the maneuver, $t_f$, that time is added to the initial Universal Time Coordinated (UTC), $UTC_0$, to produce the final UTC, $UTC_f$. Given $UTC_f$, Vallado's algorithm 14, *JulianDate*, calculates the corresponding Julian Date (JD), or $JD_f$. Given $JD_f$, Vallado's algorithm 29, *sun*, calculates the position vector from the Earth to the Sun, $r_{e2s}$, in the J2000 ECI frame in astronomical units (AUs). This is converted to the distance unit of choice, and then following Schaub [87], the vector from the RSO to

the Sun, $r_{c2s}$, in the RSO LVLH frame is found by:

$$r^R_{c2s} = {}^R[ON]^I(r^I_{e2s} - r^I_{e2c}), \tag{234}$$

where ${}^R[ON]^I$ is the direction cosine matrix transforming a vector from the inertial frame, $I$, to the relative frame, $R$, and is constructed with the knowledge of the RSO position and velocity vectors, $r^I_{e2c}$ and $v^I_{e2c}$, at the final time, $t_f$. The first two elements of $r^R_{c2s}$ are then extracted and normalized to produce the projected 2-D unit vector pointing from the RSO to the Sun, or $\hat{r}^R_{c2s_{proj}}$.

Given $\hat{r}^R_{c2s_{proj}}$, the inspector satellite must enter the prescribed NMC at the $x$ and $y$ location along the NMC that intersects this projected vector. The mission planner must prescribe the type of NMC to enter, by defining the LROEs $a_e$, $y_{d_0}$, $z_{max}$, and $\gamma$. It must be noted that the values chosen for $a_e$ and $y_{d_0}$ must be chosen such that the RSO is inside of the NMC, otherwise the opportunity to circumnavigate the RSO and remain in phase with the sunlight vector doesn't exist. The closer $y_{d_0}$ is to zero, the better the motion of the inspector satellite will stay in phase with the projected sunlight vector.

This approach assumes it is appropriate to use the projection of the Sun vector. Thus, the orbit properties of the RSO and the time of year must combine to produce a small angle between the RSO orbit plane and the Sun vector for the scenario of interest. For example, the approximate best case, where the projected Sun vector is the same as the actual Sun vector, is the case where $i = 23.5^o$ (or $i = -23.5^o$) and $\Omega = 0^o$ (or $\Omega = 180^o$) at any time of the year. Table 6 shows several best case scenarios for when there is no difference between the actual and projected Sun vectors. Again, the goal is for the actual Sun vector to be as close as possible to parallel with the orbital plane of the RSO. In addition to the RSO orbit properties and the time of year, an appropriate value for $z_{max}$ (not too large compared to $a_e$) must be chosen

to ensure that the motion of the inspector satellite remains close to the orbital plane of the RSO. For cases where there is an angle between the actual and projected Sun vector, a value for $z_{max}$ could be prescribed to correct for that angle, but only at one point during the period. At the opposite side of the period, the non-zero value for $z_{max}$ would increase (worsen) the angle between the line of sight vector from the inspector satellite to the RSO and the vector from the Sun to the RSO.

**Table 6. Example Cases for No Difference Between Actual and Projected Sun Vector**

| $i$ (deg) | $\Omega$ (deg) | Time of Year |
|---|---|---|
| 23.5 | 0 | Any |
| -23.5 | 180 | Any |
| Any | 0 or 180 | Vernal & Autumnal Equinoxes |
| 90 | 90 or 270 | June & December Solstices |

To find the $x$ and $y$ location for entry into the NMC, first the in-plane angle to $\hat{r}^R_{c2s_{proj}}$ is calculated,

$$\alpha_t = \text{atan2}(\hat{r}^R_{c2s_{proj}}[2], \hat{r}^R_{c2s_{proj}}[1]), \tag{235}$$

where the second and first components of $\hat{r}^R_{c2s_{proj}}$ are used in MATLAB's *atan2* function respectively. For an NMC to exist, $x_d$ must be equal to zero, which is equivalent to the constraint shown in Equation 82. This means that $y_d = y_{d_0}$ via Equation 71. Plugging $y_{d_0}$ into Equation 78, solving for $\sin\beta$, and substituting it into Equation 77 generates the following relationship,

$$\dot{x} = (y - y_{d_0})\frac{\omega}{2}. \tag{236}$$

Denoting the target state as $X_t = [x_t, y_t, z_t, \dot{x}_t, \dot{y}_t, \dot{z}_t]$ and given $\alpha_t$,

$$y_t = x_t \tan\alpha_t. \tag{237}$$

Substituting Equation 237 into Equation 236, and then substituting the resulting equation along with Equation 82 into the equation for $a_e$, Equation 162, the solution for $x_t$ can be obtained:

$$x_t = \frac{-\tan\alpha_t y_{d_0} \pm \sqrt{\tan\alpha_t^2 a_e^2 - 4y_{d_0}^2 + 4a_e^2}}{\tan\alpha_t^2 + 4}. \tag{238}$$

Using the $x$ component of $\hat{r}^R_{c2s_{proj}}$, a simple quadrant check can be performed to obtain the correct value for $x_t$. Then, the rest of the in-plane target states can be found by evaluating Equations 82, and 237 before 236. The in-plane target phasing angle, $\beta_t$, can then be calculated using Equation 60, and given $z_{max}$ and $\gamma$, the out-of-plane target states can be calculated via Equations 80–81.

For the hard sunlight constraint, the inspector is constrained to enter the NMC exactly at this target state, $X_t$. For the soft sunlight constraint, the inspector may enter the NMC at any $\beta \in [\beta_t \pm \theta_s]$. This means that the final state after the burn-coast-burn sequence, $X_f$, given by Equations 228–233, can vary along the trajectory in the prescribed range for $\beta$.

### 4.3.5 Initial Guess Methods.

#### 4.3.5.1 CW Targeting.

Three new methods are introduced to produce better initial guesses than MATLAB's PSO and thus improve the probability of convergence of the NLP solver. The first is CW targeting, which uses the HCW STM, Equation 51, a given maneuver time, $t_f$, and a desired or target position and velocity, $p_t$ and $v_t$ respectively, to find the $\Delta V$ magnitude and direction for the two burns in the burn-coast-burn sequence. These impulsive burns are given by Equations 52 and 56. Once $\Delta V_1$ and $\Delta V_2$ are

calculated, the initial guess is then produced by using the relationship

$$Fd_b = m||\Delta V||_2, \tag{239}$$

where the force, $F$, imparted by the thruster(s) in the direction of the burn and the mass of the satellite, $m$, are assumed to be constant for the duration of the burn, $d_b$. Thus, using an estimate for $F$ and $m$, or $a_0 = \frac{F}{m}$, the duration of a burn is estimated to be

$$d_b = \frac{||\Delta V||_2}{a_0}, \tag{240}$$

for the first and second burns in the burn-coast-burn sequence, using $\Delta V_1$ and $\Delta V_2$ respectively, where $a_0$ can be updated to $a_{0_2}$ for the second burn if desired. Then, the initial guess for the NLP solver can be generated, where the control optimization variables, $[\alpha_1, \phi_1, t_{c_f}, \alpha_2, \phi_2, t_{2_f}]$, can be calculated from the vectors $\Delta V_1$, $\Delta V_2$, and from $d_{b1}$, $d_{b2}$, and $t_f$. For the hard sunlight constraint, the initial guess contains just those calculated control optimization variables. For the soft sunlight constraint, the initial guess for the additional optimization variable $\beta$ is calculated using Equation 60, which is a function of the calculated target states. One major advantage of this initial guess method is that it is deterministic, unlike the next two methods. However, the larger the calculated impulsive burns are, the greater the error will be when transforming them to actual finite burns for the initial guess.

#### 4.3.5.2 Modified MATLAB PSO.

The second new initial guess method developed is a modification to MATLAB's PSO. One of the features of MATLAB's PSO is that it has an adaptive neighborhood size and inertia weight for the particles at each iteration. Also, when a component

of a particle's position vector violates an upper or lower bound, that optimization variable is moved back to the bound it violated, and the corresponding component in the velocity vector is set to zero. The modified PSO, developed herein to produce better initial guesses, follows MATLAB's PSO, except for the logic used when an element of a particle's position vector reaches a bound. Now, instead of bringing a variable back to the bound it violated, if the variable that violated a bound is an angle variable, i.e. $\alpha_1$, $\phi_1$, $\alpha_2$, or $\phi_2$, then it is transported to the opposite bound and its velocity remains as it was. This modification helps to account for the periodic nature of the angle variables, without removing the periodic bounds (which would create additional problems).

#### 4.3.5.3 MATLAB's GA.

The third new initial guess method used is MATLAB's GA which has built-in ways to handle nonlinear constraints, unlike MATLAB's PSO where it can be difficult to minimize the performance index and satisfy constraints simultaneously. The two options for handling nonlinear constraints are the *Augmented Lagrangian Genetic Algorithm* and the *Penalty Algorithm* where the former is the default option. According to the MATLAB documentation pages, with the *Penalty Algorithm*, MATLAB's GA solves the problem by first attempting to create a feasible GA population with respect to all constraints via its *fmincon* algorithm by starting from a variety of initial points from within the bounds. It automatically uses the tournament selection type, and then proceeds with the normal algorithm, using the penalty function as the fitness measure. This means that if an individual in the population is feasible, then the penalty function is the fitness function. If an individual is infeasible, then the penalty function is the maximum fitness function from the feasible individuals in the population plus the sum of the constraint violations of the current individual. With

both methods tested, the non-default option, *Penalty Algorithm*, performs better for the problems considered, and is thus chosen to satisfy the nonlinear constraints, with all the other GA settings left as their defaults.

### 4.3.6 Optimization Problem Formulations.

#### 4.3.6.1 Parameter Optimization Problem.

The parameter optimization problem for the metaheuristic initial guess methods and the NLP solver is formulated in two different ways, depending on which sunlight constraint is enforced. For the hard sunlight constraint, the optimization variables are

$$\chi = [\alpha_1, \phi_1, t_{c_f}, \alpha_2, \phi_2, t_{2_f}], \tag{241}$$

thus $n = 6$, where the 1 and 2 subscripts for the angles are for burns one and two, $t_{2_f}$ is the fraction of the final time, $t_f$, when burn two starts, and $t_{c_f}$ is the fraction of the time up until the burn two start time when coasting begins (or burn one ends). The box constraints, or simple bounds on the optimization variables, do not limit the search space in any way, and are:

$$\chi_l = \left[0, -\frac{\pi}{2}, 0, 0, -\frac{\pi}{2}, 0\right] \tag{242}$$

$$\chi_u = \left[2\pi, \frac{\pi}{2}, 1, 2\pi, \frac{\pi}{2}, 1\right]. \tag{243}$$

The objective is to minimize the fuel used and thus the sum of the durations of burn one and burn two. The objective can be written in terms of the optimization variables

as:

$$J = t_{2_f}(t_{c_f} - 1). \tag{244}$$

The problem is subject to the $m = 6$ equality constraints:

$$h = X_f - X_t = [x_f - x_t, y_f - y_t, z_f - z_t, \dot{x}_f - \dot{x}_t, \dot{y}_f - \dot{y}_t, \dot{z}_f - \dot{z}_t]^T, \tag{245}$$

where the gradient of the objective function is simply:

$$\nabla J(\chi) = [0, 0, t_{2_f}, 0, 0, t_{c_f} - 1]^T, \tag{246}$$

and the Hessian of the objective function is:

$$\nabla^2 J(\chi) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{247}$$

For the soft sunlight constraint, there is one more optimization variable,

$$\chi = [\alpha_1, \phi_1, t_{c_f}, \alpha_2, \phi_2, t_{2_f}, \beta], \tag{248}$$

where $\beta$ is bounded as follows to allow the inspector to enter within some angle of $\beta_t$:

$$\chi_l = \left[0, -\frac{\pi}{2}, 0, 0, -\frac{\pi}{2}, 0, \beta_t - \theta_s\right] \tag{249}$$

$$\chi_u = \left[2\pi, \frac{\pi}{2}, 1, 2\pi, \frac{\pi}{2}, 1, \beta_t + \theta_s\right]. \tag{250}$$

The objective is the same, but the $m = 6$ equality constraints now become equivalent to those used in Problem B-1, Equation 218, but for the prescribed NMC,

$$h = \begin{bmatrix} x_f - \left(\frac{-a_e}{2}\cos\beta\right) \\ y_f - (a_e \sin\beta + y_{d_0}) \\ z_f - (z_{max}\sin(\gamma+\beta)) \\ \dot{x}_f - \left(\frac{a_e}{2}\omega\sin\beta\right) \\ \dot{y}_f - (a_e\omega\cos\beta) \\ \dot{z}_f - (z_{max}\omega\cos(\gamma+\beta)) \end{bmatrix} = 0, \tag{251}$$

where $X_f = [x_f, y_f, z_f, \dot{x}_f, \dot{y}_f, \dot{z}_f]$ again comes from Equations 228–233. The gradient of the objective function and the Hessian of the objective function are slightly modified, with a zero row added to the end of the gradient, and a row and column of zeros added to the bottom and right-hand side of the Hessian of the objective function.

The PSO algorithms and the GA of course do not use any derivative information and attempt to satisfy constraints by appending them to the cost function or by using built-in methods. The main advantage of producing an initial guess with these methods is that they are global methods, and may be more likely to find the global minimum. Also, as has been mentioned, no numerical integration is required due to the development of the analytic propagation of the burn-coast-burn sequence, and thus they are computationally fast. The solution from these methods then serves as an initial guess for MATLAB's *fmincon* and/or GPOPS-II.

Given an initial guess, the NLP solver either approximates derivative information or uses user-supplied derivatives, which in this case can be exact analytic derivatives. With exact derivatives, an NLP solver may perform better than if approximate derivatives are used.

#### 4.3.6.2 Analytic Derivatives.

With an analytic expression for the six HCW states after the burn-coast-burn sequence, the gradient of the objective function, the Jacobian of the constraint vector, and the Hessian of the Lagrangian can be calculated analytically and supplied to NLP solvers to potentially increase convergence rates, the accuracy of the solution, and/or computation time. This has been accomplished by using MATLAB's Symbolic Toolbox. The gradient of the objective function, $\nabla J$, is an $n \times 1$ vector where $n$ is the number of optimization variables, which for this subproblem is 6 or 7 depending on which two of the sunlight constraints is enforced, and was presented in Equation 246. The Jacobian of the equality constraint vector, $h$, is represented by $\nabla h$ and has the gradient of each constraint along the columns of the $n \times m$ Jacobian matrix. The Hessian, $H$, is the $n \times n$ matrix of second derivatives of the Lagrangian, $\mathcal{L}$:

$$H = \nabla^2_{\chi\chi}\mathcal{L}(\chi, \lambda) = \nabla^2 J(\chi) + \sum_{r=1}^{m} \lambda_i \nabla^2 h_i(\chi), \tag{252}$$

where $\lambda$ is the vector of Lagrange multipliers. The only derivative information explicitly shown here is $\nabla J(\chi)$, Equation 246, and the Hessian of the objective, or $\nabla^2 J(\chi)$, Equation 247. The Jacobian matrix of the constraint vector, $\nabla h$, and the Hessian of the Lagrangian, $H$, are far too lengthy to present on paper, but have been successfully computed using MATLAB's Symbolic Toolbox.

#### 4.3.6.3 GPOPS-II Problem Formulation.

If higher fidelity is desired, the solution from the initial guess methods or the NLP solver may be supplied as the initial guess for GPOPS-II, where this higher-fidelity model uses more accurate equations of motion and accounts for continuous mass loss. The GPOPS-II problem formulation is split up into three phases, similar to Problem B-1, where the states and times are linked between each of the phases, and the phases correspond to the burn-coast-burn sequence where each burn is a constant direction burn, and the acceleration magnitude increases with mass loss. Thus, it is a three-phase optimal control problem. The bounds on the thrust angles, the objective function, and the constraints are the same as in the previous section for the two types of sunlight constraints, but the final state after the burn-coast-burn sequence, $X_f$, is not calculated analytically, but found via GPOPS-II using the higher-fidelity equations of motion. Because of this, the calculated analytic derivatives cannot be supplied to the solver. The resulting NLP generated by GPOPS-II is then solved by either IPOPT or SNOPT. This high-fidelity model is valid for low and high-thrust engines since it accounts for continuous mass loss, and also good for longer distance maneuvers to inject the inspector satellite into an NMC, since the CNERMs are being used. A summary of the solvers used in the mid and high-fidelity models and their validity is shown in Table 7.

**Table 7. Mid and High-Fidelity Models Comparison**

| Solver | Equations of Motion | Suitable For: |
|---|---|---|
| PSO, GA | HCW | -Initial guess |
| NLP (*fmincon*) | HCW | -Low-thrust, Close prox-ops |
| GPOPS-II | CNERMs | -Low or high thrust, Close or long distance maneuvers |

### 4.3.7 Simulation and Results with Sunlight Constraints.

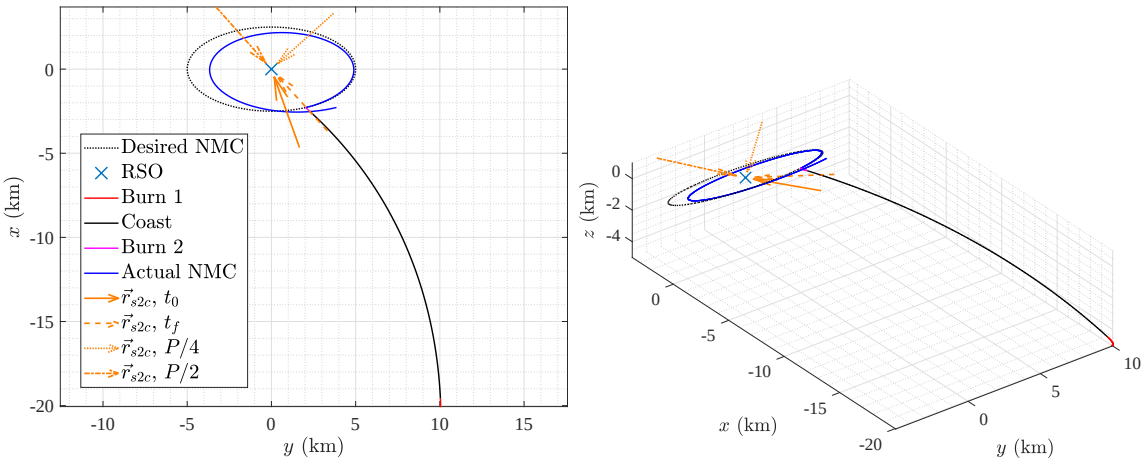
Multiple scenarios were developed and simulated with the parameters shown in Table 8.

**Table 8. Problem B-2 Simulation Parameters**

| Sun and Time | RSO Properties | Inspector Properties | NMC Parameters |
|---|---|---|---|
| $\theta_s = 45^o$ | $a_c = 42,164.137$ km | $a_0 = 0.02$ N/kg | $a_e = 5$ km |
| $\text{Year}_0 = 2017$ | $r^I_{e2c_0} = [a_c; 0; 0]$ km | $c = 3.33$ km/s | $y_{d_0} = 0$ km |
| $\text{Month}_0 = \text{Aug}$ | $v^I_{e2c_0} = [0; 3.0747; 0]$ km/s | $x_0 = -20$ km | $z_{max} = 1$ km |
| $\text{Day}_0 = 31$ | | $y_0 = 10$ km | $\gamma = 90^o$ |
| $\text{Hr}_0 = 23$ | | $z_0 = -5$ km | |
| $\text{Min}_0 = 0$ | | $\dot{x}_0 = -1.5$ m/s | |
| $\text{Sec}_0 = 0$ | | $\dot{y}_0 = 0.4$ m/s | |
| | | $\dot{z}_0 = 1.1$ m/s | |

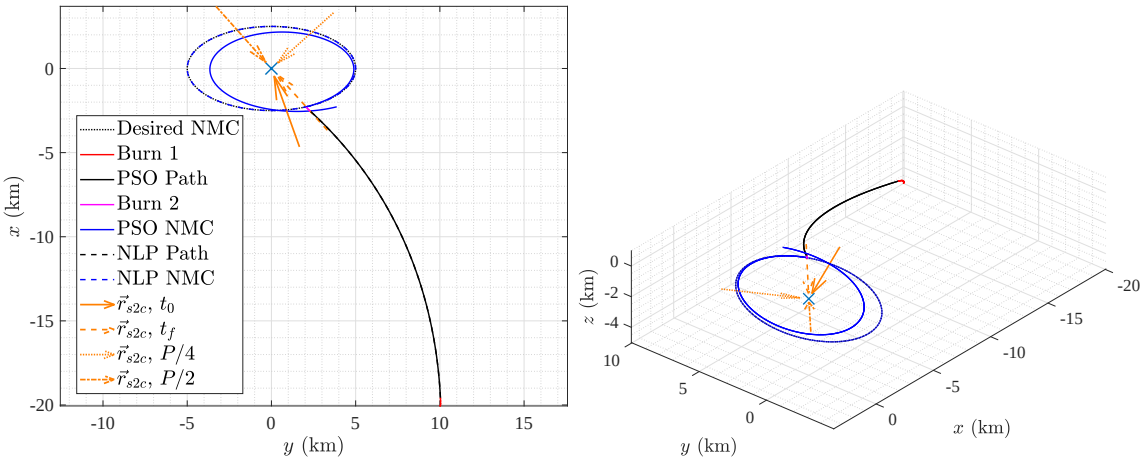
The first simulation result presented is a typical solution obtained with the PSO, although each PSO solution may vary due to the inherent stochastic nature of the algorithm. The number of particles used was 300, with weights, $W_r$, chosen judiciously for the appended cost function $\tilde{J}(\chi)$ to minimize constraint violations. This simulation was first run with the hard sunlight constraint and a fixed final time of 1.5 hours with results shown in Figure 24. The PSO solution does well at lining up with the projected sunlight vector at the final time, but does not produce the exact desired NMC. However, it may be adequate to use as the initial guess for *fmincon* or GPOPS-II. The computation time for this typical solution is 6.16 seconds, taking advantage of the analytic propagation of the burn-coast-burn sequence. The actual objective value (opposed to the appended objective value) is 502.45 seconds, representing the total engine-on time for the minimum-fuel maneuver. However, as mentioned, the constraints were not met to the desired tolerance.

A PSO solution is then provided as the initial guess to MATLAB's *fmincon*, to see if the NLP solver can better satisfy constraints and further refine and minimize the total engine-on time. For now, derivative information is not supplied and *fmincon* approximates the derivatives via a finite difference method. Figure 25 shows the




(a) Orbital Plane View (b) 3-D View

**Figure 24. Problem B-2, PSO Solution, Hard Sunlight Constraint, $t_f = 1.5$ hrs**

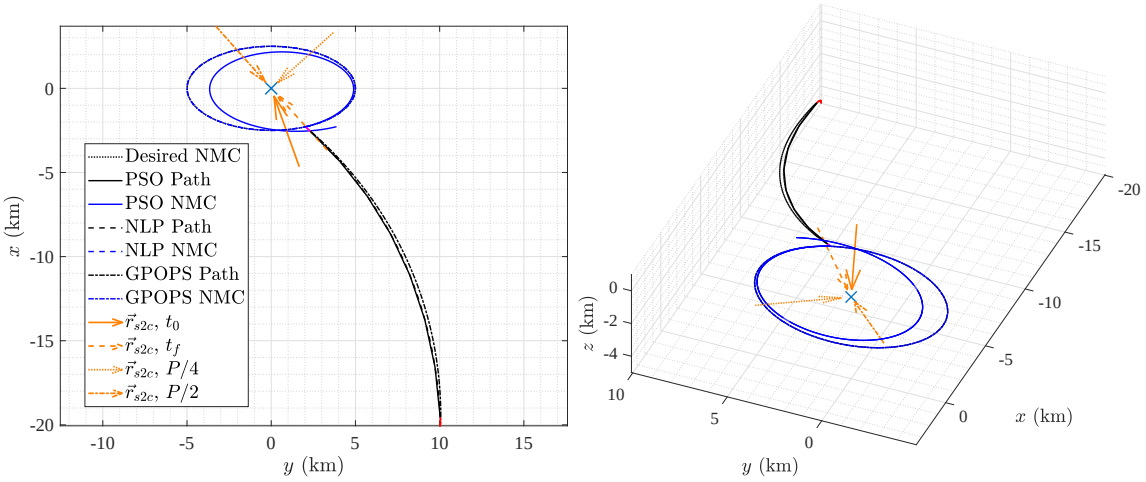



(a) Orbital Plane View (b) 3-D View

**Figure 25. Problem B-2, Mid-Fidelity Solutions, Hard Sunlight Constraint, $t_f = 1.5$ hrs**

PSO solution with the NLP solution overlaid, demonstrating how the NLP solver successfully injects the inspector satellite into the desired NMC. The computation time for the NLP solution is only 0.0105 seconds where the *fmincon* algorithm used in this case was *sqp*. Given the initial guess and the analytic propagation of the maneuver, the computation time is extremely fast. The exit flag was the best possible, and the objective value actually increased a very small amount, to 502.95 seconds, allowing the constraints to be met unlike the PSO solution. The best possible exit

flag is a 1, with the four possible exit flags from MATLAB's *fmincon* being −2, 0, 1 or 2, defined as follows: −2 — the step size is below the tolerance and constraints are not satisfied; 0 — the maximum number of function evaluations or iterations has been reached; 1 — a local minimum has been found; and 2 — the step size is below the tolerance and constraints have been satisfied (local minimum possible).




(a) Orbital Plane View (b) 3-D View

**Figure 26. Problem B-2, All Model Solutions, Hard Sunlight Constraint, $t_f = 1.5$ hrs**

A typical PSO solution is then provided as an initial guess to GPOPS-II, where the high-fidelity model uses the initial guess from the mid-fidelity model to generate a solution using the CNERMs and accounting for continuous mass loss. The GPOPS-II solution is noticeably different than the PSO and NLP solutions and is a higher-fidelity solution for guidance. Derivative information is not supplied to GPOPS-II and it numerically approximates the derivatives using central differencing. For this run, SNOPT is the solver of choice, and GPOPS-II solves the problem in 1.234 seconds given the PSO solution as an initial guess, with the best possible exit flag. The engine-on time increases to 510.77 seconds in the high-fidelity solution. It must be noted that the targeted NMC is an HCW NMC, i.e. it only exists perfectly in the realm of the HCW equations. Thus, the resulting natural motion for the GPOPS-II

solution was numerically propagated for one period, with MATLAB's *ode45*, and as can be seen in Figure 26, the resulting natural motion is very close to the desired NMC, meaning that the constraints were sufficient in this case to produce the desired trajectory.

The same set of simulations were performed again, but now with the soft sunlight constraint enforced. In order to see how the objective value changes by allowing a margin, $\theta_s$, on either side of $\beta_t$, the same initial conditions and problem parameters were used as before, seen in Table 8. The PSO, NLP, and GPOPS-II solutions are all shown together in Figures 27-28. As expected, the inspector satellite enters the NMC at a different location compared to when the hard sunlight constraint is enforced, entering close to (within $\theta_s$ of $\beta_t$) the sunlight vector. This allows for more fuel savings, as shown in Table 9, which summarizes the results for all three solvers under both of the sunlight constraints. The exit flags are shown, where the exit flag codes from MATLAB's *fmincon* are used for GPOPS-II exit flags, i.e., IPOPT and SNOPT exit flags are converted to the equivalent *fmincon* exit flags. Again, a flag of 2 means that a local minimum is possible, a 1 means that a local minimum has been found, a 0 means that the maximum number of iterations or function evaluations has been reached, and a $-2$ means that the size of the current step is less than the tolerance but constraints are not satisfied to the desired tolerance. As seen in Table 9, there is one instant where the *sqp* exit flag is a $-2$. The solution is still close and appears correct, as shown in Figure 27, but technically cannot be denoted as a local minimum. The results shown in Table 9 were obtained with no derivative information supplied to the solvers. For the aforementioned case when an exit flag of $-2$ was obtained, the exit flag actually changed to a 1 when the derivative of the objective function and the Jacobian of the constraints were supplied to the solver. However, it is not reported as such in the table and remains as a $-2$ to be consistent with no derivative information

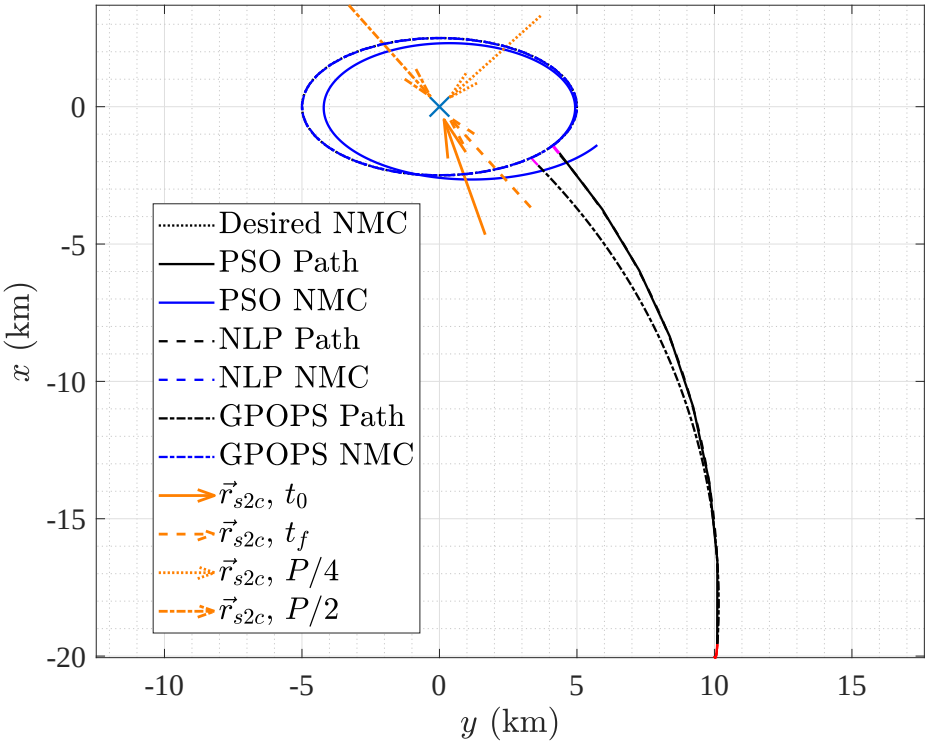



**Figure 27. Problem B-2, All Solutions, Soft Sunlight Constraint, 2-D, $t_f = 1.5$ hrs**

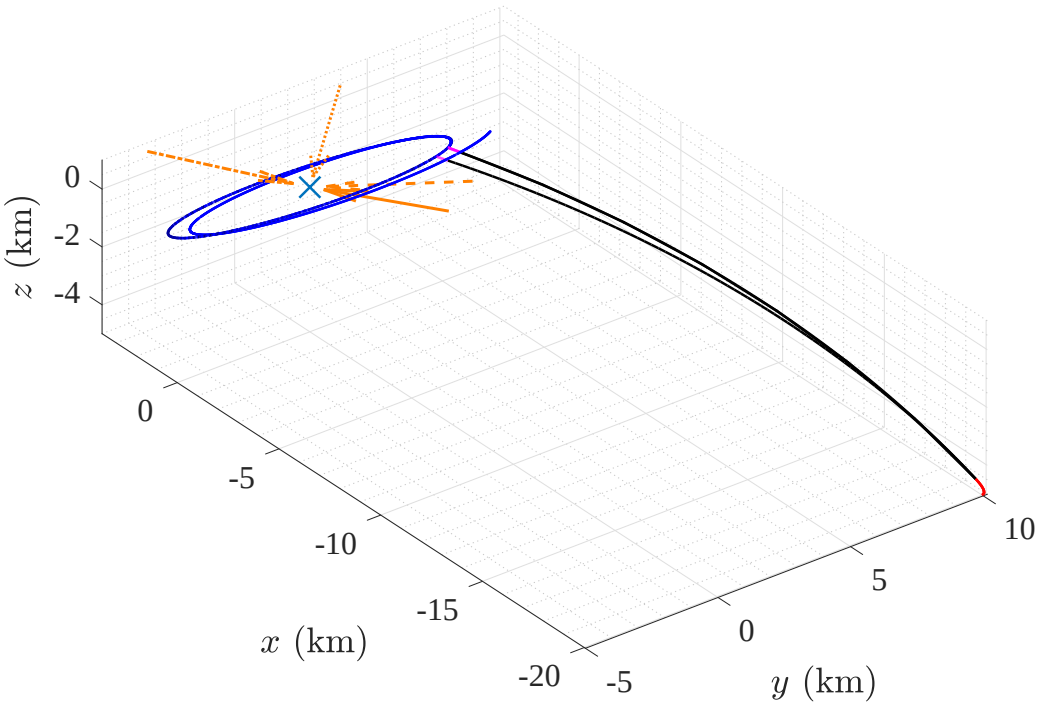



**Figure 28. Problem B-2, All Solutions, Soft Sunlight Constraint, 3-D, $t_f = 1.5$ hrs**

being supplied to the solvers for these results. The simulations shown in Table 9 (and throughout Problem B-2) used the default settings for the PSO and NLP solver, with the exception of any explicitly mentioned settings and increasing the maximum number of iterations and/or function evaluations. Regarding GPOPS-II, no special treatment was required to obtain the solutions.

**Table 9. Problem B-2 Simulation Results, No Derivative Information Supplied**

| Solver | Sunlight Constraint | Exit Flag | $J$ (s) | CPU Time (s) |
|---|---|---|---|---|
| PSO | Hard | NA | 502.45 | 6.16 |
| | Soft | NA | 494.45 | 7.56 |
| NLP (*fmincon sqp*) | Hard | 1 | 502.95 | 0.0105 |
| | Soft | $-2$ | 494.39 | 0.0269 |
| GPOPS-II (SNOPT) | Hard | 1 | 510.77 | 1.234 |
| | Soft | 1 | 501.15 | 1.749 |

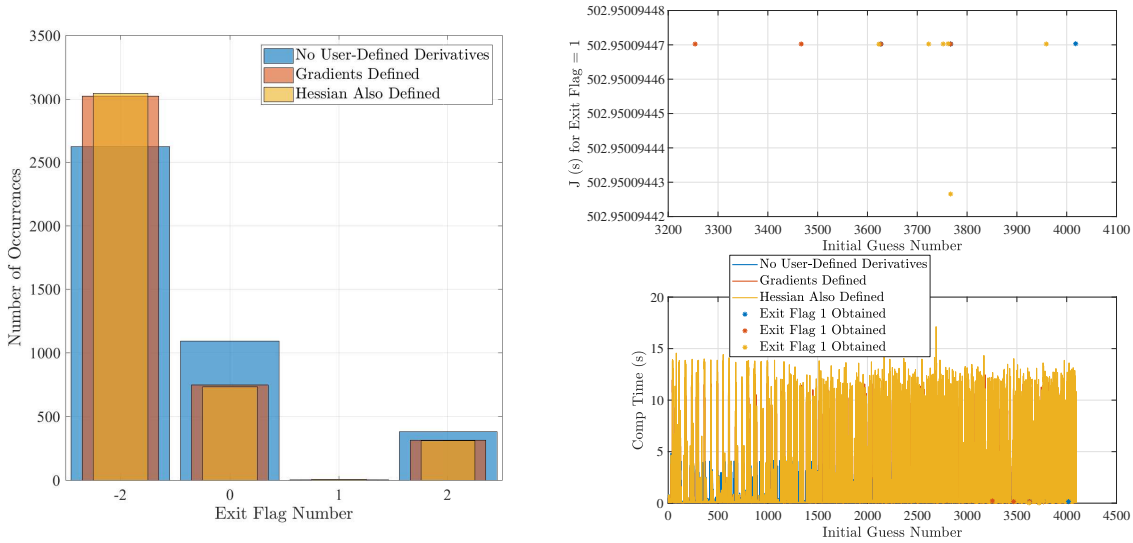
### 4.3.8 Initial Guess and NLP Performance with Sunlight Constraints.

As mentioned earlier, supplying derivative information to an NLP solver may aid in obtaining a solution and increase the speed and accuracy of the solution. Also, the solution obtained depends on the initial guess provided, which depending on the problem may need to be a very good initial guess in order to obtain the desired solution. In addition, depending on the problem, an interior point method may perform better than a sequential quadratic programming method, and vice versa. Thus, a sensitivity analysis for convergence has been developed, to determine how well the NLP solver performs given the initial guess method, the type of sunlight constraint, the NLP algorithm, and the level of derivatives supplied by the user, for the same parameters as seen in Table 8.

To begin, a baseline is generated by providing uninformed initial guesses to the NLP solver, where the initial guesses come from a hypergrid of the optimization variables. That is, a grid is generated for each optimization variable, made up of

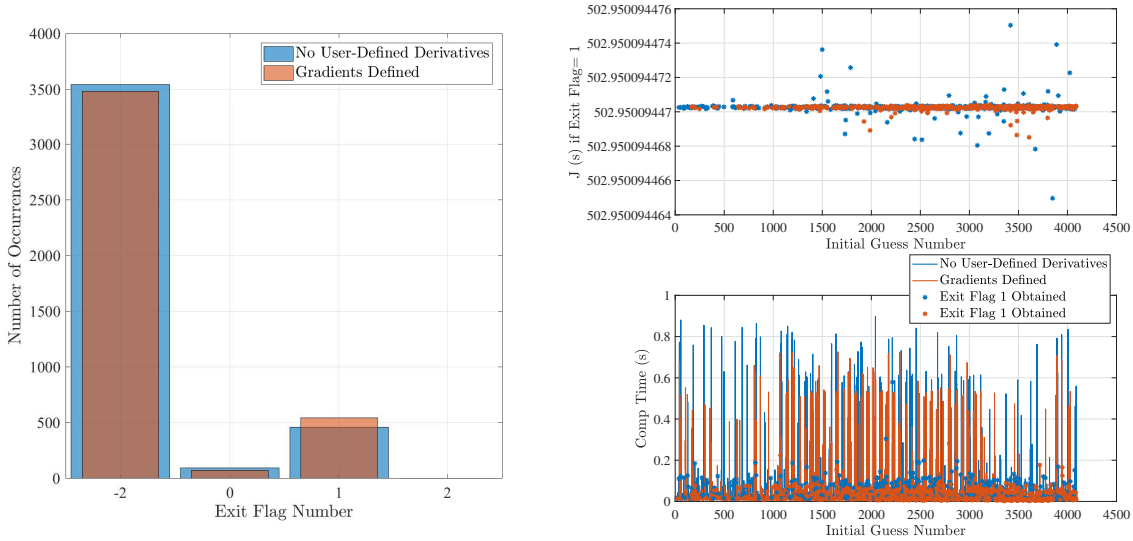
$N$ equally spaced points from its lower bound to its upper bound. For example, if $N = 3$, then the variable $\alpha_1$ would have possible values 0, $\pi$, and $2\pi$ for the set of initial guesses. Therefore, each possible value of each optimization variable forms an initial guess with every other possible combination of the other optimization variables. Thus, there are $N^n$ initial guesses, where $n = 6$ for the hard sunlight constraint, and $n = 7$ for the soft sunlight constraint. Two sensitivity analyses for convergence are then performed for this first case with uninformed initial guesses, one for each type of sunlight constraint.

For the hard sunlight constraint, $N$ was chosen to be 4, and thus $4^6 = 4{,}096$ initial guesses were generated to analyze the performance of the NLP solver for the hard sunlight constraint. Due to time constraints, $N$ was limited to 4, and it must be noted that the initial guesses do span the space, but do not have a high enough resolution for one initial guess to happen to fall on a solution, unless extremely lucky. Each initial guess was supplied to *fmincon* under five different conditions: 1) Using *interior-point* with no derivative information supplied, 2) using *interior-point* with the gradient of the objective function and the Jacobian of the constraint vector supplied, 3) using *interior-point* with the Hessian of the Lagrangian also supplied, 4) using *sqp* with no derivative information supplied, and 5) using *sqp* with the gradient of the objective function and the Jacobian of the constraint vector supplied. Note that *sqp* does not have the option of supplying the Hessian of the Lagrangian. Thus, the NLP solver was run $N^n \times 5 = 20{,}480$ times. For each run, the exit flag, the objective value, the computation time, and the optimization variables were collected. The results from the sensitivity analysis for the hard sunlight constraint can be seen in Figure 29 for *interior-point* and Figure 30 for *sqp*. Tabulated results are also provided in Table 10 where the top half is for the hard sunlight constraint, showing the exit flag percentages and also the average computation time for each exit flag.




(a) Exit Flag Occurrences (b) CPU Time and Successful J Values

**Figure 29. NLP Performance, *interior-point*, Grid Initial Guesses, Hard Sun Constraint**

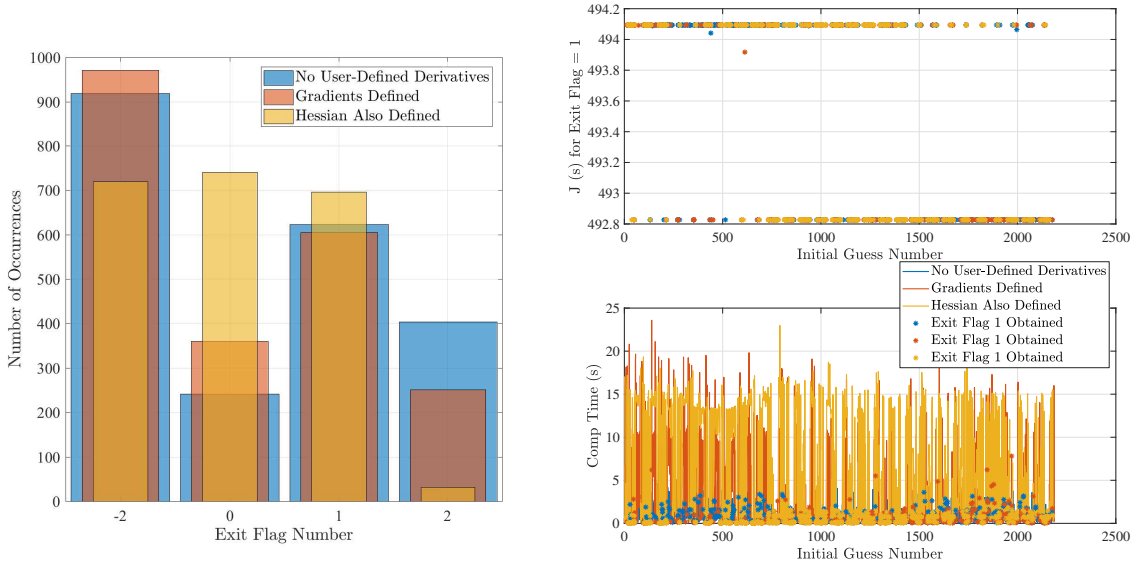



(a) Exit Flag Occurrences (b) CPU Time and Successful J Values

**Figure 30. NLP Performance, *sqp*, Grid Initial Guesses, Hard Sunlight Constraint**

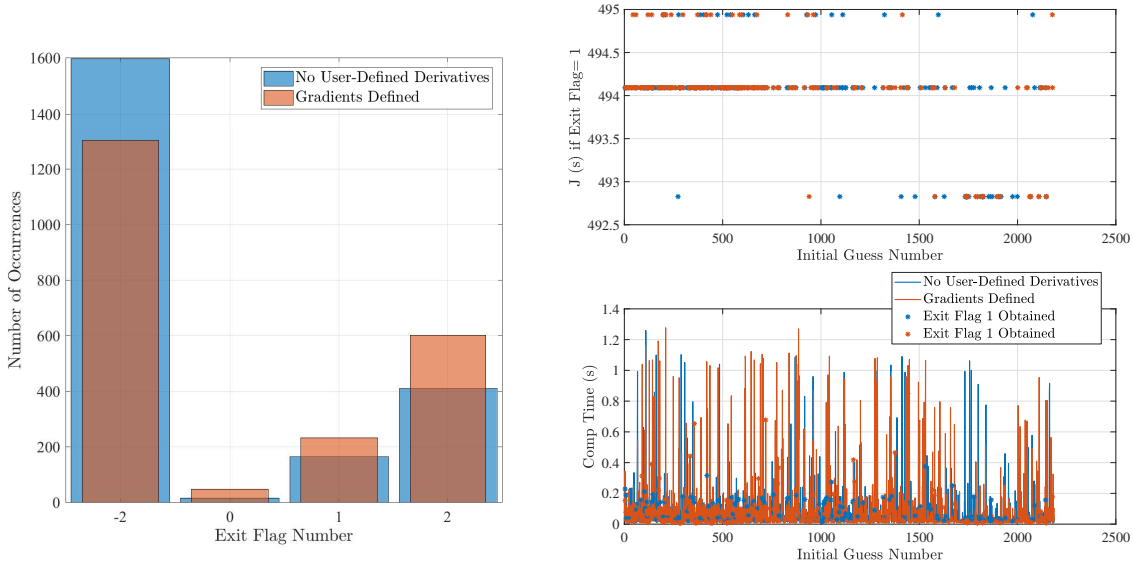
Before extracting results from the hard sunlight constraint data, the results are also presented for the soft sunlight constraint sensitivity analysis. For these results, $N = 3$, $n = 7$, and thus there were $3^7 = 2{,}187$ initial guesses supplied in the same manner as for the hard sunlight constraint sensitivity analysis. The results from

the sensitivity analysis for the soft sunlight constraint can be seen in Figure 31 for *interior-point* and Figure 32 for *sqp*. Numerical results are also provided in Table 10 along with the hard sunlight constraint sensitivity analysis data.




(a) Exit Flag Occurrences (b) CPU Time and Successful J Values

**Figure 31. NLP Performance, *interior-point*, Grid Initial Guesses, Soft Sun Constraint**




(a) Exit Flag Occurrences (b) CPU Time and Successful J Values

**Figure 32. NLP Performance, *sqp*, Grid Initial Guesses, Soft Sunlight Constraint**

**Table 10. NLP Performance, Grid Initial Guesses, Both Sunlight Constraints**

| Algorithm | Supplied | Exit Flag Obtained (%) | | | | Mean CPU Time (s) for Exit Flag | | | |
|---|---|---|---|---|---|---|---|---|---|
| Hard Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 63.96 | 26.66 | 0.07 | 9.30 | 0.4184 | 3.4118 | 0.1536 | 0.0706 |
| int-pt | Gradients | 73.90 | 18.31 | 0.10 | 7.69 | 0.6113 | 9.8143 | 0.1678 | 0.0362 |
| int-pt | Hessian Also | 74.37 | 17.90 | 0.15 | 7.59 | 0.7487 | 11.7051 | 0.1593 | 0.0404 |
| sqp | None | 86.52 | 2.27 | 11.21 | 0 | 0.0271 | 0.6706 | 0.0542 | NA |
| sqp | Gradients | 85.01 | 1.73 | 13.26 | 0 | 0.0320 | 0.5573 | 0.0289 | NA |
| Soft Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 42.02 | 11.01 | 28.48 | 18.47 | 0.5702 | 3.6766 | 0.6465 | 1.0049 |
| int-pt | Gradients | 44.39 | 16.46 | 27.66 | 11.47 | 1.9488 | 11.4112 | 0.4521 | 0.4007 |
| int-pt | Hessian Also | 32.87 | 33.83 | 31.82 | 1.46 | 3.1247 | 13.9998 | 0.4320 | 0.6606 |
| sqp | None | 73.02 | 0.73 | 7.49 | 18.74 | 0.0360 | 1.0392 | 0.0854 | 0.1354 |
| sqp | Gradients | 59.62 | 2.19 | 10.65 | 27.52 | 0.0405 | 1.0005 | 0.0759 | 0.1131 |

The same two analyses were performed again for the hard and soft sunlight constraints, but this time each initial guess came from a MATLAB PSO solution instead of from the grid. Thus, 4,096 PSO solutions were generated for the hard constraint case, and 2,187 PSO solutions were generated for the soft constraint case. The results are presented in Table 11, to be compared to the results in Table 10.

**Table 11. NLP Performance, MATLAB PSO Initial Guesses, Both Sunlight Constraints**

| Algorithm | Supplied | Exit Flag Obtained (%) | | | | Mean CPU Time (s) for Exit Flag | | | |
|---|---|---|---|---|---|---|---|---|---|
| Hard Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 46.50 | 2.51 | 0 | 50.97 | 0.1897 | 4.0820 | NA | 0.0151 |
| int-pt | Gradients | 47.29 | 1.83 | 0 | 50.87 | 0.2507 | 12.0366 | NA | 0.0120 |
| int-pt | Hessian Also | 47.36 | 1.78 | 0 | 50.85 | 0.3184 | 14.6955 | NA | 0.0143 |
| sqp | None | 50.61 | 0.02 | 49.36 | 0 | 0.0146 | 0.6744 | 0.0061 | NA |
| sqp | Gradients | 50.58 | 0.02 | 49.38 | 0 | 0.0190 | 0.7205 | 0.0043 | NA |
| Soft Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 18.74 | 5.12 | 51.16 | 24.96 | 0.9262 | 3.4230 | 0.4692 | 1.1284 |
| int-pt | Gradients | 19.47 | 5.98 | 72.88 | 1.64 | 1.2010 | 11.0311 | 0.4451 | 1.4826 |
| int-pt | Hessian Also | 14.12 | 12.66 | 67.12 | 6.08 | 1.9800 | 13.1942 | 0.3905 | 0.6729 |
| sqp | None | 22.17 | 0.04 | 74.57 | 3.20 | 0.0360 | 1.1042 | 0.0136 | 0.1059 |
| sqp | Gradients | 16.87 | 0.54 | 76.31 | 6.26 | 0.0278 | 0.9584 | 0.0087 | 0.1002 |

Several conclusions can be drawn from the sensitivity analyses for convergence by comparing the results when the initial guesses came from the grid vs. when the initial guesses came from MATLAB's PSO:

Given the initial guesses from the grid:

- For the hard sunlight constraint, *sqp* performs best, and for the soft sunlight constraint, *interior-point* performs best, where 'best' means that the referenced method converged to an exit flag of 1 more often than the other method.

- Overall, supplying derivative information assists in obtaining a solution. For both hard and soft sunlight constraints, and for both algorithms, the percentage of exit flags equal to 1 increased as the level of derivative information supplied increased, with one exception: For the soft sunlight constraint and *interior-point*, the amount of exit flags equal to 1 decreased slightly with gradient information, but then increased to above the original percentage when the analytic Hessian was supplied.

- For every exit flag of 1 achieved, the initial guess was infeasible and uninformed, simply taken from the poor-resolution set of initial guesses, showing that the algorithms are capable at times to find the desired solution even with a very bad initial guess.

- At best, the hard sunlight constraint scenario has a 13.26% chance of converging to an exit flag of 1, and the soft sunlight constraint scenario has a 31.82% chance of converging to an exit flag of 1, given the crude initial guesses. Thus, obviously, a good initial guess should be supplied to increase the reliability and accuracy of the developed algorithms.

- The objective value is about the same for any exit flag 1 achieved when compared to other objective values with the same sunlight constraint. This shows that for this scenario, multiple local minimums where an exit flag of 1 was achieved were not found.

- Overall, *sqp* has a faster computation time for successful results, an order of magnitude faster than *interior-point*.

Given the initial guesses from MATLAB's PSO:

- For the hard sunlight constraint, *sqp* again performs best. For the soft sunlight constraint, *sqp* actually performs slightly better than *interior-point*.

- Given the PSO initial guesses, supplying derivative information doesn't have as much of an effect on the NLP performance. For the hard sunlight constraint, it didn't result in *interior-point* in finding a solution, and slightly increased *sqp*'s performance. For the soft sunlight constraint, supplying derivatives helped, but it's interesting to note that the gradients were more helpful than the Hessian for *interior-point*.

- Given the PSO solutions as an initial guess, the hard sunlight constraint scenario's convergence percentage increased from 13.26% to 49.38%. The soft sunlight constraint scenario's convergence percentage increased from 31.82% to 76.31%. Thus, the PSO initial guess increases the convergence percentage of the NLP solver to an exit flag of 1. This gives importance to the developed PSO algorithm, and the capability it has to quickly produce a good initial guess. However, even more reliable initial guess methods are desired.

- For cases where an exit flag of 1 was achieved, the average computation time was faster given the PSO initial guesses, especially for *sqp*, about an order of magnitude faster given the PSO initial guesses as compared to successful computation times given the initial guesses from the grid.

To show how the two new metaheuristic initial guess methods yield improved convergence results, the same type of analysis is performed again for both the modified

MATLAB PSO and MATLAB's GA. Thus, the same simulation parameters are used as in Table 8. The results for both the hard and soft sunlight constraints where the solutions from the modified PSO are given as the initial guesses for the NLP solver are shown in Table 12, and the results given the MATLAB GA solutions for initial guesses are shown in Table 13.

**Table 12. NLP Performance, Modified MATLAB PSO Initial Guesses, Both Sunlight Constraints**

| Algorithm | Supplied | Exit Flag Obtained (%) | | | | Mean CPU Time (s) for Exit Flag | | | |
|---|---|---|---|---|---|---|---|---|---|
| Hard Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 17.99 | 3.98 | 0.24 | 77.78 | 0.2551 | 3.8062 | 0.1550 | 0.0182 |
| int-pt | Gradients | 18.65 | 3.44 | 0.39 | 77.51 | 0.4198 | 11.1668 | 0.0429 | 0.0157 |
| int-pt | Hessian Also | 18.70 | 3.49 | 1.12 | 76.68 | 0.5099 | 13.2954 | 0.0580 | 0.0183 |
| sqp | None | 17.80 | 0.17 | 82.03 | 0 | 0.0303 | 0.6533 | 0.0078 | NA |
| sqp | Gradients | 17.63 | 0.15 | 82.23 | 0 | 0.0300 | 0.6596 | 0.0050 | NA |
| Soft Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 3.57 | 0.27 | 75.08 | 21.08 | 0.3547 | 3.4614 | 0.5597 | 0.6749 |
| int-pt | Gradients | 2.38 | 0.27 | 94.56 | 2.79 | 1.1508 | 11.5545 | 0.3468 | 0.1680 |
| int-pt | Hessian Also | 1.87 | 0.91 | 63.33 | 33.88 | 1.9514 | 13.9828 | 0.2294 | 0.1569 |
| sqp | None | 43.16 | 0.32 | 53.91 | 2.61 | 0.0195 | 0.9723 | 0.0097 | 0.1371 |
| sqp | Gradients | 6.08 | 36.12 | 55.24 | 2.56 | 0.0756 | 0.7111 | 0.0064 | 0.0988 |

Given the initial guesses from the Modified PSO:

- For the hard sunlight constraint, *sqp* performs best.
- For the soft sunlight constraint, *interior-point* performs best.
- Supplying derivative information helps in all cases where a solution is found, except for when including the Hessian also for the soft sunlight constraint when using the *interior-point* method. However, by supplying first derivative information for the soft sunlight constraint when using the *interior-point* method, the convergence percentage increases by almost 20%.
- Given the modified PSO as an initial guess, the convergence percentage for the hard sunlight constraint increases from 49.38% from the original PSO to 82.23%.

- Given the modified PSO an an initial guess, the convergence percentage for the soft sunlight constraint increases from 76.31% from the original PSO to 94.56%.

**Table 13. NLP Performance, MATLAB GA Initial Guesses, Both Sunlight Constraints**

| Algorithm | Supplied | Exit Flag Obtained (%) | | | | Mean CPU Time (s) for Exit Flag | | | |
|---|---|---|---|---|---|---|---|---|---|
| Hard Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 0 | 0 | 98.66 | 1.34 | NA | NA | 0.0060 | 0.0150 |
| int-pt | Gradients | 0 | 0 | 99.85 | 0.15 | NA | NA | 0.0053 | 0.0129 |
| int-pt | Hessian Also | 0 | 0 | 99.85 | 0.15 | NA | NA | 0.0070 | 0.0172 |
| sqp | None | 0 | 0 | 100 | 0 | NA | NA | 0.0048 | NA |
| sqp | Gradients | 0 | 0 | 100 | 0 | NA | NA | 0.0037 | NA |
| Soft Sunlight Constraint | | -2 | 0 | 1 | 2 | -2 | 0 | 1 | 2 |
| int-pt | None | 0 | 0 | 98.45 | 1.56 | NA | NA | 0.0677 | 0.1203 |
| int-pt | Gradients | 0 | 0 | 100 | 0 | NA | NA | 0.0551 | NA |
| int-pt | Hessian Also | 0 | 0 | 100 | 0 | NA | NA | 0.0629 | NA |
| sqp | None | 3.66 | 0 | 0.46 | 95.88 | 0.0101 | NA | 0.0060 | 0.0049 |
| sqp | Gradients | 3.34 | 0.59 | 1.01 | 95.06 | 0.0885 | 0.6106 | 0.0042 | 0.0117 |

Given the initial guesses from MATLAB's GA (see Table 13):

- For the hard sunlight constraint, *sqp* performs best, with a 100% convergence rate. *interior-point* had similar results, with a 99.85% convergence rate.

- For the soft sunlight constraint, *interior-point* performs best, with a 100% convergence rate, and *sqp* doesn't perform well at all.

- Supplying derivative information either helped a small amount or had no effect on achieving convergence.

The average computation times for each metaheuristic initial guess method is shown in Table 14, emphasizing how computationally fast the metaheuristic methods are, due to the analytic propagation of the burn-coast-burn sequence and the relatively small amount of optimization variables.

The CW Targeting initial guess method is deterministic as compared to the metaheuristic initial guess methods and thus cannot be analyzed in the exact same manner.

**Table 14. Average CPU Times (seconds) for Metaheuristic Initial Guess Methods**

| Sunlight Constraint | PSO | Modified PSO | GA |
|---|---|---|---|
| Hard | 5.23 | 2.80 | 6.01 |
| Soft | 8.36 | 1.86 | 8.32 |

For the hard sunlight constraint and with $t_f = 1.5$ hours, the CW Targeting initial guess provides a successful guess only for the two *sqp* methods. For the soft sunlight constraint, it does not provide a successful initial guess for any of the five *fmincon* methods. This is due to the fixed final time, $t_f$, being too small and thus the error due to the impulsive approximation is too large. For example, if $t_f$ is changed to 3 hours, then all three *interior-point* methods converge to a solution when given the CW Targeting initial guess. This makes sense, as it's obvious that as the fixed final time increases, the required engine-on time typically decreases, and the closer the CW Targeting solution resembles the actual finite-burn solution.

To summarize these analyses, the two new metaheuristic initial guess methods developed in this subproblem perform better than MATLAB's PSO. If MATLAB's Optimization Toolbox is available, then MATLAB's GA is the method of choice, since it performs extremely well by using the non-default *Penalty Algorithm* method to handle the nonlinear equality constraints. If MATLAB's Optimization Toolbox is unavailable or not desirable for use, then the Modified PSO performs well for initial guess generation, and is relatively easy to code. If the fixed final times of the maneuvers of interest are large enough compared to the actual finite-duration burns required to implement the calculated $\Delta V$s, then CW Targeting is an extremely fast and simple method to produce a good initial guess.

### 4.3.9 Earth and Moon Field-of-View Constraints for NMC.

An inspector satellite may be equipped with a sensor which needs to be pointed at the RSO throughout the NMC. It is thus desirable that the view of the RSO be

unobscured by any bright objects behind it and in the field of view of the sensor. Therefore, this subsection develops methods to ensure the Earth and Moon remain at a prescribed angle away from the sensor boresight vector so that they do not appear in the background during the NMC.

The Earth is excluded from the background of the RSO by prescribing the appropriate kind of NMC which the inspector satellite may enter. Similar to the NMC design for Problem A, an exclusion cone with a half cone angle $\theta$ is created such that the sensor boresight vector avoids the edge of the Earth in addition to the prescribed exclusion angle for the Earth, $\theta_e$:

$$\theta = \tan^{-1}\left(\frac{R_e}{2a}\right) + \theta_e, \tag{253}$$

where $R_e$ is the radius of the Earth and $a$ is the semi-major axis of the RSO. With this half cone angle, the LROE $z_{max}$, which is the maximum out-of-plane amplification, can be calculated as

$$z_{max} = \frac{a_e}{2}\tan(\theta), \tag{254}$$

where $a_e$ is the prescribed size of the NMC. The LROE $\gamma$, which is the constant phase difference between the in-plane and out-of-plane motion, must be

$$\gamma = \pm 90^o. \tag{255}$$

If these LROE values of $z_{max}$ and $\gamma$ are chosen, then the edge of the Earth will stay at an angle $\theta_e$ away from the sensor boresight vector while pointed at the RSO throughout the NMC.

In order to keep the Moon excluded from the background, the strategy developed

is to project the Moon vector into the orbital plane of the RSO, as was done with the Sun vector, and check for cases where the in-plane angle to the projected Moon vector, $\alpha_m$, is within the Moon exclusion angle, $\theta_m$, from the expected in-plane angle to the inspector satellite (plus $180^o$), $\alpha_d$, throughout the NMC. In order to do this, the first step is to use the Julian Date at the final time of the maneuver to calculate the Moon vector using Vallado's algorithm 31, *Moon* [13], following which the vector from the RSO to the Moon is calculated in the same fashion as was done for the Sun vector. The angle to this vector projected into the orbital frame of the RSO, $\alpha_m(t)$, is then calculated over the course of one period. Using the LROEs, the in-plane angle to the inspector satellite (plus $180^o$), $\alpha_d(t)$, is also calculated over the course of one period, and the difference between the two angles is calculated. The minimum difference between the two angles is extracted, $\alpha_{min}$, and then used as follows: For the hard sunlight constraint, if $\alpha_{min} < \theta_m$, then the desired NMC entry state, $X_t$, is shifted in the smallest direction away from the projected Sun vector to a position on the NMC such that the new $\alpha_d(t)$ creates a new $\alpha_{min}$ which doesn't violate $\theta_m$. For the soft sunlight constraint, the bounds on $\beta$ are changed from the original angular margin allowed from the projected Sun vector such that no $\beta$ can be chosen by the optimizer which makes $\alpha_{min}$ violate $\theta_m$. If this changes the bounds on $\beta$ enough to where $X_t$ is no longer within the bounds, then the CW Targeting target position and velocity are changed to the state on the new bound closest to the original $X_t$. The change in $X_t$ for the hard sunlight constraint and the changes on the bounds for $\beta$ for the soft sunlight constraint ensure that the Moon will not come within $\theta_m$ of the sensor boresight vector throughout the NMC, at least for the course of one period.

### 4.3.10 Simulations and Results with Multiple Lighting Constraints.

This section presents simulation results where varying types of lighting constraints are enforced for the motion in the resulting NMC. The first simulation only enforces the hard sunlight constraint, with the same parameters as shown in Table 8, except for the starting UTC date is changed to September 5th. The solution is obtained by using CW Targeting for the initial guess and then IPOPT as the NLP solver (with default settings), where both levels of analytic derivatives are supplied to IPOPT. For this case, IPOPT successfully converges to a solution unlike *fmincon* did previously with $t_f = 1.5$ hrs. The resulting NMC from CW Targeting is erroneous, but an adequate initial guess for IPOPT, as shown in Figure 33, where the projected sunlight vectors from the Sun to the RSO, $r_{s2c}$, are shown at the beginning of the maneuver, $t_0$, the final time of the maneuver, $t_f$, and at one-fourth and one-half of the way through the NMC, measured by the RSO period, $P$.

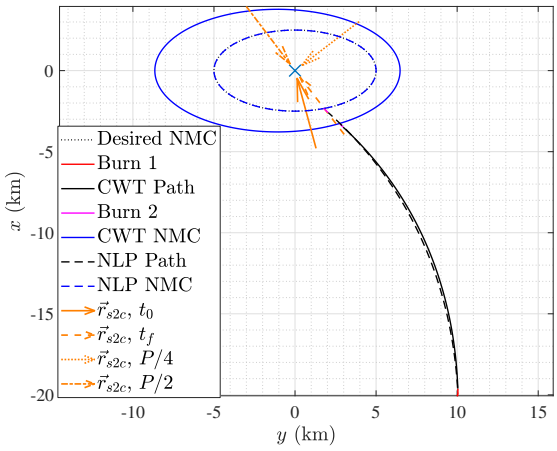



**Figure 33. Problem B-2, CWT, IPOPT Solutions, Hard Sun Constraint, $t_f = 1.5$ hrs**

For the new UTC date of September 5th, the angle to the projected Moon vector, $\alpha_m$, comes within $\theta_m$ of $\alpha_d$, and even crosses $\alpha_d$, as seen in Figure 34. This means that if the inspector satellite enters the NMC at the original $X_t$ (aligned with the Sun vector), then there is a good chance that at some point during the NMC, no

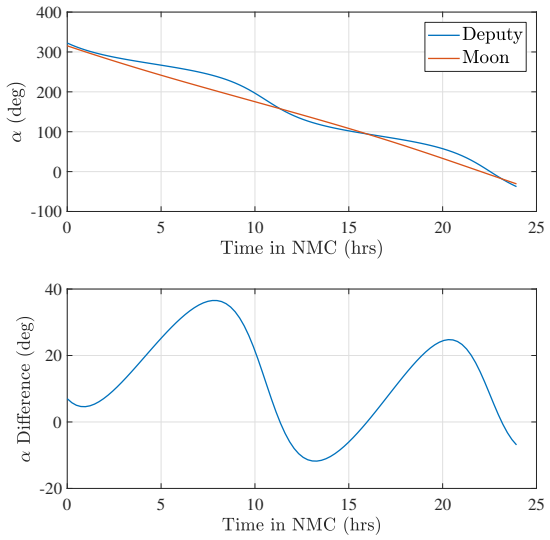
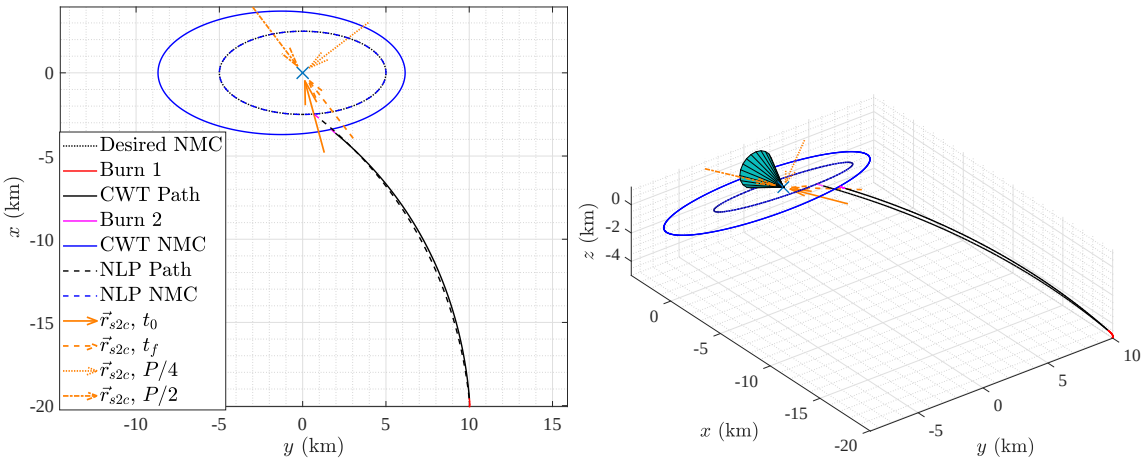



**Figure 34. Problem B-2, Moon Conflict, Hard Sunlight Constraint, $t_f = 1.5$ hrs**

matter the NMC cross-track properties, the Moon will appear in the background, within $\theta_m$ of the sensor boresight vector. Thus, for this scenario, the minimum shift for $\alpha_d$ must be $21.78^o$, which is the minimum angle $\alpha_m$ must be shifted to not cross $\alpha_d$ ($\alpha_{min} = 11.78^o$), and obtain $\theta_m$ which for this case is $10^o$. This means that the inspector must enter the NMC at a shifted $X_t$ which puts the inspector satellite behind the Sun vector at the time of entry, as seen in Figure 35 (a), as compared to Figure 33.

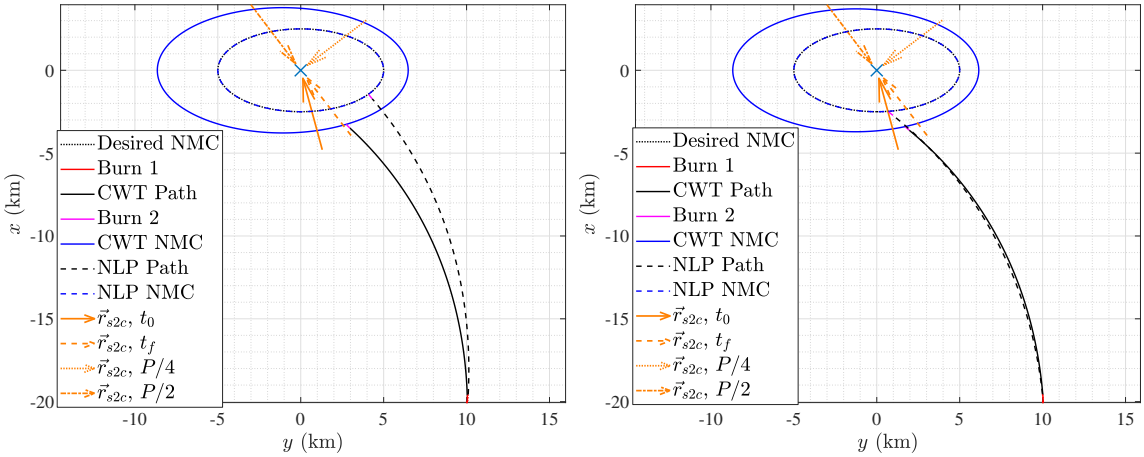
The Earth lighting constraint has also been enforced, by prescribing the appropriate LROEs $z_{max}$ and $\gamma$ as given by Equations 254–255. The result can be seen in Figure 35 (b), where the exclusion cone shows how the NMC must be designed such that the Earth does not appear in the field of view throughout the NMC.

The same parameters are used again but this time the soft sunlight constraint is enforced. Figure 36 (a) shows the solution where just the soft sunlight constraint is enforced, where the entry point has shifted from $X_t$ (or $\beta_t$) to within $\pm\theta_s$ to further minimize fuel usage. Now, if the Moon lighting constraint is also enforced, the shifted $X_t$ (and the corresponding $\beta_t$) from the hard sunlight constraint example now becomes




(a) $X_t$ Shifted to Avoid Moon Conflict (b) NMC Designed to Avoid Earth Conflict

**Figure 35. B-2, CWT, NLP Solutions, Hard Sun Constraint, Field-of-View Constraints, $t_f = 1.5$ hrs**




(a) Moon Conflict (b) Entry Shifted to Avoid Moon Conflict

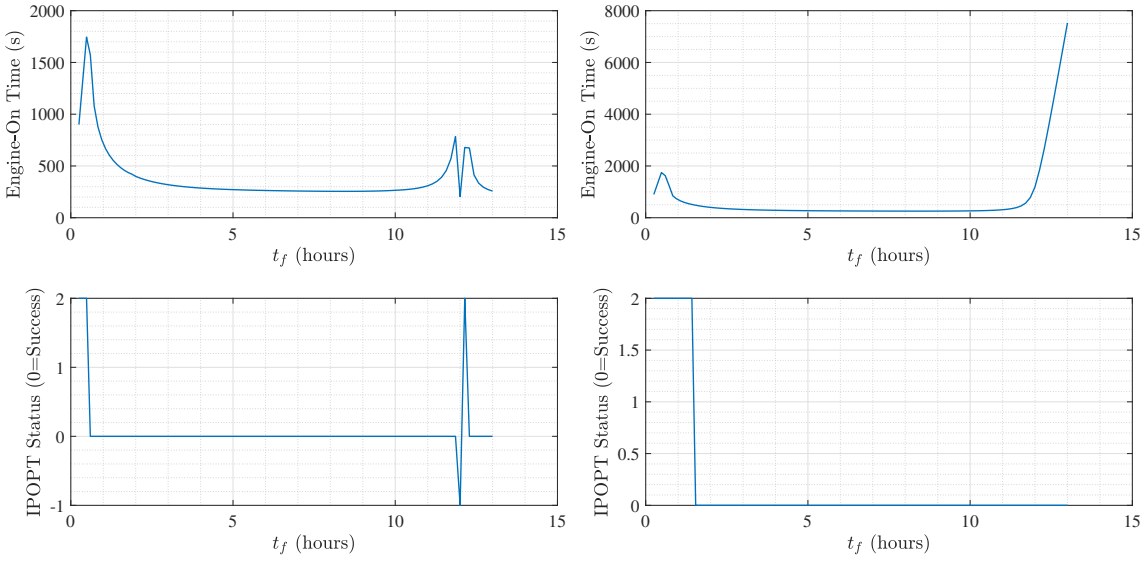
**Figure 36. Problem B-2, CWT, NLP Solutions, Soft Sunlight Constraint, $t_f = 1.5$ hrs**

one of the new bounds for $\beta$. Since in this example $X_t$ was shifted behind the original $X_t$, this puts a new upper bound on $\beta$. Thus, the original lower bound is unchanged, and the new upper bound ensures that the inspector satellite will not enter the NMC such that the Moon may appear in the field of view. The result can be seen in Figure 36 (b), where the entry point into the NMC has shifted to the optimal entry point in the new allowable range for $\beta$, which happens to be at the shifted $X_t$ from the hard sunlight constraint case.

#### 4.3.10.1 Range of Optimal Solutions.

Mission planners may desire a range of optimal solutions based on varying fixed final times in order to evaluate the trade-offs for how much fuel could be saved by changing the fixed final time of the maneuver. Typically, this is a monotonically decreasing curve, where the amount of fuel required decreases as the fixed final time increases. However, with unique constraints such as the developed lighting constraints, this monotonic behavior may not always be the case. To produce a range of optimal solutions, two methods have been developed. The first is to use CW Targeting as the initial guess for IPOPT, where both the initial guess and NLP solver is executed for each fixed final time. The second is to use a homotopic approach, where the IPOPT solution to the original fixed final time problem is used as the initial guess for the next closest fixed final time problem, also solved with IPOPT. This process then continues, step by step, for the rest of the fixed final times. An example range of solutions has been found for $t_f \in [0.25, 13]$ hours with the soft sunlight constraint and Earth lighting constraint enforced. Figure 37 (a) shows the range of optimal solutions where the first approach is used, and Figure 37 (b) shows the range of optimal solutions when the second approach is used. It is interesting to note that the CW Targeting approach provides better initial guesses for IPOPT for decreasing fixed final times compared to the homotopy approach. It is also interesting to note that the CW Targeting approach has some convergence issues around $t_f = 12$ hours, but regains convergence afterwards, whereas the homotopic approach successfully converges, but follows a local minimum to very high engine-on times as the fixed final time increases. Figure 38 shows the region of interest where the detected minimum lies, where a fixed final time of about 8.5 hours produces the lowest engine-on time solution of about 255 seconds for this scenario. The effects of choosing the correct fixed final time are apparent. By examining a range of optimal solutions for a given scenario, mission planners may

choose the correct one, and mission longevity can be significantly impacted.




(a) CW Targeting (b) Homotopy

**Figure 37. Problem B-2, Range of Optimal Solutions, Soft Sunlight Constraint, Costs & Exit Flags**

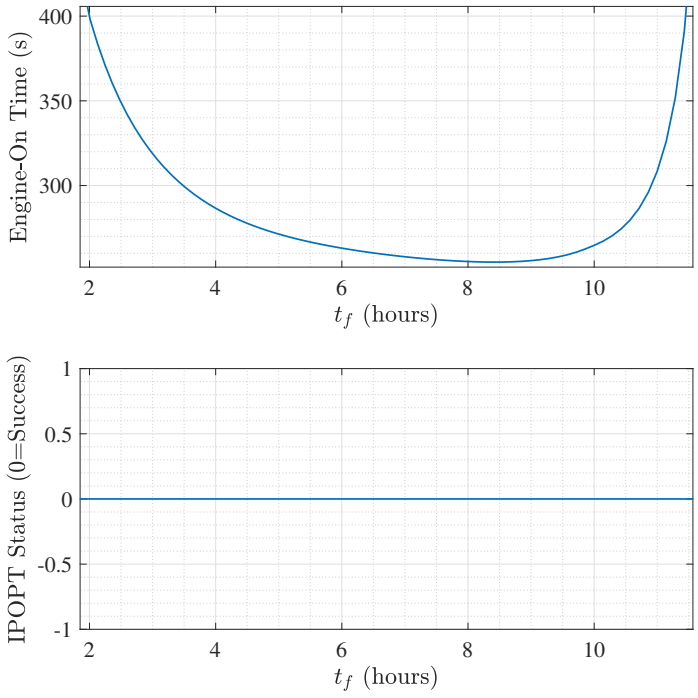



**Figure 38. Problem B-2, Range of Optimal Solutions, Soft Sunlight Constraint, Costs & Exit Flags, Zoomed In**

### 4.3.11 Problem B-2 Conclusion.

Improved mid and high-fidelity models have been developed to find fuel-optimal guidance for an inspector satellite with on/off thrusters to maneuver into an NMC with favorable lighting conditions. An analytic expression for the HCW states after a burn-coast-burn sequence, as a function of the minimum required control variables, can be developed and used to generate analytic derivatives for use in an NLP solver. Regarding the initial guess for the NLP solver, it has been shown that a modified MATLAB PSO and MATLAB's GA perform better than MATLAB's PSO and are computationally fast, where the GA is especially reliable. CW Targeting can also be used to generate an initial guess for longer fixed final time problems, and has the advantage of being analytic and deterministic as compared to the metaheuristic methods. These improved initial guess methods help lead to fast computation times by the NLP solvers as well. The NLP solvers' performance can also be enhanced by providing the exact analytic derivatives, where it has been shown that supplying first derivative information may significantly improve performance. Via the analyses performed, it is recommended that the *sqp* algorithm be used for the hard sunlight constraint, and the *interior-point* algorithm be used for the soft sunlight constraint, and that the analytic first derivative information be supplied in both cases. With the initial guess and NLP formulations denoted as the mid-fidelity model, their solution can then be supplied to a higher-fidelity model, a pseudospectral solver, where the CNERMs can be used to produce a more accurate, high-fidelity solution. With the mid-fidelity solution used as an initial guess, the high-fidelity model is able to quickly converge as well, making both the mid and high-fidelity models computationally efficient. Lighting constraints can also be successfully incorporated into the optimization problem. The hard sunlight constraint successfully constrains the satellite to enter the NMC such that it is between the RSO and the Sun, aligned with the

projected sunlight vector upon entry. The soft sunlight constraint allows a margin to the hard sunlight constraint, and further minimizes fuel usage. And finally, field-of-view constraints can be developed to successfully keep the Earth and Moon out of the inspector's field of view throughout the motion in the NMC. A range of optimal solutions can also be generated with the developed methods to provide mission planners with options. Thus, both a mid and high-fidelity model have been improved to generate fuel-optimal guidance, where the mid-fidelity model is valid for low-thrust engines and close proximity operations, and the high-fidelity model is valid for low and high-thrust engines as well as longer-distance injection maneuvers.

## 4.4 Problem B-3

### 4.4.1 Overview.

Problem B-3 considers using a coast-burn-coast-burn sequence instead of just a burn-coast-burn sequence, in order to save more fuel in certain scenarios and add more flexibility to adhere to more constraints. Like Problem B-1, Problem B-3 investigates minimum-fuel and minimum-time maneuvers into a relative teardrop trajectory, but now subject to lighting and collision constraints. Thus, the same type of lighting constraints as were developed in Problem B-2 for maneuvers into an NMC will be developed for maneuvers into a teardrop, and in addition to these lighting constraints, passive and active collision constraints will also be developed. Therefore, the first part of this subproblem presents the coast-burn-coast-burn sequence. Next, the targeted teardrop is explained in further detail, and then the sunlight and field-of-view constraints are developed. Then, passive and active collision avoidance constraints are presented, which take advantage of the analytic propagation of the coast-burn-coast-burn sequence. The optimization problem formulations are then presented, including an explanation of the analytic derivatives. Finally, simulation results are shown, along with example ranges of optimal solutions and some solution validation.

### 4.4.2 Equations of Motion and Control Definition.

Problem B-3 uses the HCW equations of motion where again the acceleration terms for each burn are constants and are described by Equations 194–196. For the maneuvers investigated in this subproblem, a coast-burn-coast-burn sequence is allowed, instead of just a burn-coast-burn sequence.

### 4.4.3 Analytic Propagation of a Coast-Burn-Coast-Burn Sequence.

In certain scenarios, depending on the initial conditions and the fixed final time to maneuver to a target trajectory, an additional coast period before the first burn in the burn-coast-burn sequence may allow for the use of less fuel. Thus, this subsection develops the analytic propagation of a new coast-burn-coast-burn sequence where again the acceleration magnitude and direction of the burns are held constant during each burn, approximating a high-efficiency on/off thruster. The analytic propagation of the sequence is a function of variables which describe the duration of each phase and the three-dimensional direction of each of the two burns. These variables, shown in Table 15, form the minimum number of variables required to analytically propagate the coast-burn-coast-burn sequence and become the core optimization variables. Figure 39 shows how the variables relate to the transition times in the coast-burn-coast-burn sequence.

**Table 15. Coast-Burn-Coast-Burn Sequence Variables**

| Variable | Description | Bounds |
|---|---|---|
| $t_{1_f}$ | fraction of coast 2 start time to start first burn | $[0, 1]$ |
| $t_{c_f}$ | fraction of burn 2 start time to start second coast | $[0, 1]$ |
| $t_{2_f}$ | fraction of final time, $t_f$, to start second burn | $[0, 1]$ |
| $\alpha_1$ | in-plane acceleration direction for burn 1 | $[0, 2\pi]$ |
| $\phi_1$ | out-of-plane acceleration direction for burn 1 | $[-\pi/2, \pi/2]$ |
| $\alpha_2$ | in-plane acceleration direction for burn 2 | $[0, 2\pi]$ |
| $\phi_2$ | out-of-plane acceleration direction for burn 2 | $[-\pi/2, \pi/2]$ |

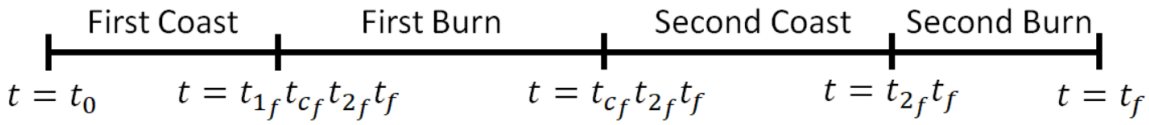



**Figure 39. Coast-Burn-Coast-Burn Sequence Parameterization**

Starting with the given time to accomplish the maneuver, $t_f$, the duration of each phase is calculated by using the first three optimization variables where $t_{1_f}, t_{c_f}, t_{2_f} \in$

$[0, 1]$. The phase durations are:

$$d_{c1} = t_{1_f} t_{c_f} t_{2_f} t_f \tag{256}$$

$$d_{b1} = t_{c_f} t_{2_f} t_f - d_{c1} = t_{c_f} t_{2_f} t_f (1 - t_{1_f}) \tag{257}$$

$$d_{c2} = t_{2_f} t_f - t_{c_f} t_{2_f} t_f = t_{2_f} t_f (1 - t_{c_f}) \tag{258}$$

$$d_{b2} = t_f - t_{2_f} t_f = t_f (1 - t_{2_f}), \tag{259}$$

representing (from top to bottom) the duration of the first coast, the duration of the first burn, the duration of the second coast, and the duration of the second burn.

Given the initial conditions, $X_0 = [x_0, y_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0]$, the analytic propagation of the first coast phase, given its duration in Equation 256, is given by the HCW STM:

$$X_{c1_f} = \Theta(t_0 + d_{c1}, t_0) X_0. \tag{260}$$

The final state from the first coast, $X_{c1_f}$, then becomes the initial condition for the first burn phase. The first burn phase, with a duration of $d_{b1}$ in Equation 257, is then

propagated analytically by Equations 197–202:

$$x_{b1_f} = \frac{1}{\omega^2}\Big(2\omega\dot{y}_{c1_f} + 4\omega^2 x_{c1_f} + \omega\dot{x}_{c1_f}\sin(d_{b1}\omega) + a_0\cos(\alpha_1)\cos(\phi_1)$$
$$- 3\omega^2 x_{c1_f}\cos(d_{b1}\omega) - 2\omega\dot{y}_{c1_f}\cos(d_{b1}\omega) - a_0\cos(d_{b1}\omega)\cos(\alpha_1)\cos(\phi_1)$$
$$- 2a_0\sin(d_{b1}\omega)\cos(\phi_1)\sin(\alpha_1) + 2a_0 d_{b1}\omega\cos(\phi_1)\sin(\alpha_1)\Big) \tag{261}$$

$$y_{b1_f} = \frac{-1}{2\omega^2}\Big(4\omega\dot{x}_{c1_f} - 2\omega^2 y_{c1_f} - 8\omega\dot{y}_{c1_f}\sin(d_{b1}\omega) + 12 d_{b1}\omega^3 x_{c1_f} + 6 d_{b1}\omega^2\dot{y}_{c1_f}$$
$$- 8a_0\cos(\phi_1)\sin(\alpha_1) - 12\omega^2 x_{c1_f}\sin(d_{b1}\omega) - 4\omega\dot{x}_{c1_f}\cos(d_{b1}\omega)$$
$$+ 8a_0\cos(d_{b1}\omega)\cos(\phi_1)\sin(\alpha_1) - 4a_0\sin(d_{b1}\omega)\cos(\alpha_1)\cos(\phi_1)$$
$$+ 3a_0 d_{b1}^2\omega^2\cos(\phi_1)\sin(\alpha_1) + 4a_0 d_{b1}\omega\cos(\alpha_1)\cos(\phi_1)\Big) \tag{262}$$

$$z_{b1_f} = \frac{1}{2\omega^2}\Big(2a_0\sin(\phi_1) - 2\cos(d_{b1}\omega)(-z_{c1_f}\omega^2 + a_0\sin(\phi_1))$$
$$+ 2\omega\dot{z}_{c1_f}\sin(d_{b1}\omega)\Big) \tag{263}$$

$$\dot{x}_{b1_f} = \frac{1}{\omega}\Big(2\omega\dot{y}_{c1_f}\sin(d_{b1}\omega) + 2a_0\cos(\phi_1)\sin(\alpha_1) + 3\omega^2 x_{c1_f}\sin(d_{b1}\omega)$$
$$+ \omega\dot{x}_{c1_f}\cos(d_{b1}\omega) - 2a_0\cos(d_{b1}\omega)\cos(\phi_1)\sin(\alpha_1)$$
$$+ a_0\sin(d_{b1}\omega)\cos(\alpha_1)\cos(\phi_1)\Big) \tag{264}$$

$$\dot{y}_{b1_f} = \frac{-1}{\omega}\Big(3\omega\dot{y}_{c1_f} + 6\omega^2 x_{c1_f} + 2\omega\dot{x}_{c1_f}\sin(d_{b1}\omega) + 2a_0\cos(\alpha_1)\cos(\phi_1)$$
$$- 6\omega^2 x_{c1_f}\cos(d_{b1}\omega) - 4\omega\dot{y}_{c1_f}\cos(d_{b1}\omega) - 2a_0\cos(d_{b1}\omega)\cos(\alpha_1)\cos(\phi_1)$$
$$- 4a_0\sin(d_{b1}\omega)\cos(\phi_1)\sin(\alpha_1) + 3a_0 d_{b1}\omega\cos(\phi_1)\sin(\alpha_1)\Big) \tag{265}$$

$$\dot{z}_{b1_f} = \frac{1}{\omega}\sin(d_{b1}\omega)(-z_{c1_f}\omega^2 + a_0\sin(\phi_1)) + \dot{z}_{c1_f}\cos(d_{b1}\omega). \tag{266}$$

This produces the state after the first burn, $X_{b1_f}$, and the pattern continues, where the second coast phase is then propagated with Equation 260, but now with $X_{b1_f}$ as the initial condition and $d_{c2}$, Equation 258, as the duration. This produces the final

state from the second coast phase, $X_{c2_f}$. The second burn is then propagated with Equations 261–266, but now with $X_{c2_f}$ as the initial condition and $d_{b2}$, Equation 259, as the duration. This produces the final state from the second burn phase, $X_{b2_f}$. Thus, as a function of the seven core optimization variables in Table 15, an analytic expression has been developed for the final six HCW states after a coast-burn-coast-burn sequence, denoted as $X_{b2_f}$ or $X_f$:

$$X_{b2_f} = X_f = [x_f, y_f, z_f, \dot{x}_f, \dot{y}_f, \dot{z}_f] = f(t_{1_f}, t_{c_f}, t_{2_f}, \alpha_1, \phi_1, \alpha_2, \phi_2), \tag{267}$$

where the computations were facilitated by MATLAB's Symbolic Toolbox. These are the final states, $X_f$, at the given fixed final time, $t_f$, upon completing the maneuver. Note also that for the second burn, the acceleration magnitude is updated to account for mass lost during the first burn as was done previously,

$$a_{0_2} = \frac{a_0}{1 - d_{b1}\frac{a_0}{c}}, \tag{268}$$

where again $c$ is the effective exhaust velocity. The complete analytic expressions for the final states, $X_f$, are too long to show, but have been symbolically computed and used, particularly in order to compute analytic derivatives, which is discussed in a later section.

### 4.4.4 Targeted Teardrop.

The teardrop parameter equations developed by Lovell, et al. [17, 1] deserve some extra attention in order for mission planners to correctly choose the parameters which produce the desired relative motion. As Lovell's papers explain, the equations have been developed for cases where $x_d < 0$, i.e. for lower teardrops, where the cusp or intersection of the trajectory occurs below the rounded portion of the teardrop. Thus,

the following clarifications are based on the equations for a lower teardrop.

If the user wishes to prescribe the teardrop parameters $D$ and $T_p$, then these values, when used in the equations for $a_e$ and $x_d$, will ensure a lower teardrop is created, i.e. with $x_d < 0$, as long as the allowable values for $D$ and $T_p$ are chosen. These two values determine $a_e$ and $x_d$ according to Equations 85–86, repeated here for convenience:

$$a_e = \frac{6D\frac{\omega T_p}{2}}{3\frac{\omega T_p}{2} - 4\sin(\frac{\omega T_p}{2})} \tag{269}$$

$$x_d = \frac{-4D\sin(\frac{\omega T_p}{2})}{3\frac{\omega T_p}{2} - 4\sin(\frac{\omega T_p}{2})}. \tag{270}$$

Thus, there is a vertical asymptote when $\omega T_p \approx 2.55139$, or when $\frac{T_p}{P} \approx 0.4061$. Therefore, values of $T_p$ close to this asymptote should be avoided. Also, in order to generate a lower teardrop and maintain the requirement that $x_d < 0$, then there are two regions where the appropriate values for $D$ and $T_p$ must be chosen. To the left of the asymptote, where $\omega T_p \in (0, 2.55139)$, $D$ should be a negative value, and thus the point of maximum radial position will occur beneath the RSO at $D$. If values of $T_p$ are desired where $T_p \in (2.55139/\omega, P)$, then the value for $D$ must be positive and the resulting teardrop is still a lower teardrop, but with the maximum radial position occurring above the RSO at (positive) $D$ instead of beneath the RSO. In this case, as $T_p \to P$, the resulting motion approaches an NMC.

The cross-track motion for a teardrop trajectory is also of interest. Typically teardrop trajectories are created to remain in the orbit plane of the RSO. However, some type of cross-track motion may be desired. As was shown in Problem B-1, the point of maximum radial position can be chosen to coincide with $z_{max}$, positive or negative. Another cross-track motion of interest is for the maximum radial position to coincide with $z = 0$, such that during half of the teardrop the cross-track position

is negative and during the other half it is positive. Table 16 explicitly lays out options for different cross-track motions of interest and the corresponding required values for $\gamma$.

**Table 16. Teardrop Cross-Track Options**

| Desired Cross-Track Motion | $\gamma$ (deg) |
|---|---|
| Reach Positive $z_{max}$ at Maximum $x$ | -90 |
| Reach Negative $z_{max}$ at Maximum $x$ | 90 |
| Start Positive, Reach $z = 0$ at Maximum $x$ | 0 |
| Start Negative, Reach $z = 0$ at Maximum $x$ | 180 |
| Planar Teardrop (i.e. $z_{max} = 0$) | NA |

Given these clarifications, the user may correctly choose the necessary parameters to target the desired relative teardrop trajectory. The user thus needs to choose $D$ and $T_p$, which produce the required $a_e$ and $x_d$ and prescribes the in-plane geometry of the teardrop. The values for $z_{max}$ and $\gamma$ prescribe the desired cross-track geometry. The placement of the teardrop along the in-track axis is prescribed by $y_T$ which determines $y_{d0}$ via Equation 206. And finally, the point at which the satellite enters the teardrop trajectory can be prescribed by choosing $\beta_t$, or a range of allowable entry angles can be prescribed by setting bounds on $\beta$. These LROEs can then be used to calculate the Cartesian states via Equations 76–81 to be targeted by the coast-burn-coast-burn sequence.

### 4.4.5 Lighting Constraints for Teardrop.

The teardrop lighting constraints developed in this subsection are similar to those developed in Problem B-2 for entry into an NMC. The key difference here, when formulating lighting constraints for entry into a teardrop trajectory, is that the relative trajectory is unbounded. For a bounded relative trajectory, e.g. an NMC, no matter what the fixed final time is for the maneuver, the desired lighting condition can be obtained or will be obtained after a certain amount of natural motion in the NMC.

However, when entering a teardrop trajectory, the inspector satellite has a limited amount of time to maneuver into the trajectory before the desired lighting condition has passed. If the first opportunity does pass, then the spacecraft must wait longer for the next opportunity to arrive or enter the trajectory at such a point where it can coast for the amount of time required to sync with the next opportunity.

Two types of sunlight constraints, similar to those developed in Problem B-2, have been developed for entry into a teardrop trajectory. The first type is called a tight (or hard) sunlight constraint. The tight sunlight constraint ensures that the vector from the Sun to the RSO, or $r_{s2c}$, when projected into the orbital plane, is directly aligned with the projected vector from the inspector satellite to the RSO, $r_{d2c}$, when the inspector satellite reaches the maximum radial position in the teardrop. Thus, this type of lighting constraint applies to lower teardrops where $D$ is negative, and where the teardrop is being formed directly beneath the RSO and not shifted in the in-track direction ($y_T = 0$). The tight sunlight constraint is formulated by first approximating the amount of time from $t_0$, the time at the beginning of the maneuver, until the projected Sun vector is pointed in the desired direction. This is approximated by assuming that over the course of one or two days the projected sunlight vector points in the same inertial direction, and thus rotates in the relative frame equal to the constant mean motion, $\omega$. Thus, the amount of time from $t_0$ until the Sun is in the desired position is

$$\tau = \frac{\alpha_s}{\omega}, \tag{271}$$

where $\alpha_s$ is the angle to the projected Sun vector at $t_0$. It is then assumed that entry into the teardrop is desired at or before the cusp, or the intersection point. Thus, given $\tau$, there are a series of checks which need to occur.

- Check 1: If $\tau < \frac{1}{2}T_p$, then $\tau = \tau + P$. In other words, if there is not enough time

to coast from the cusp to the top of the teardrop, then the inspector satellite must wait until the next opportunity.

- Check 2: If $\tau < \frac{1}{2}T_p + t_{f_{min}}$, then $\tau = \tau + P$ (where $t_{f_{min}}$ is the minimum-time solution for the given scenario). In other words, if there is not enough time to enter the teardrop trajectory, then the inspector satellite must wait until the next opportunity.

- Check 3: If $t_f < t_{f_{min}}$, then $t_f = t_{f_{min}}$. In other words, if the user has chosen a $t_f$ smaller than the minimum-time solution, then the $t_f$ is changed and the problem is solved with the minimum-time solution.

- Check 4: If $\tau < \frac{1}{2}T_p + t_f$, then $\tau = \tau + P$. In other words, if the provided $t_f$ is too large, then the inspector satellite must wait until the next opportunity.

After all of these checks are complete and required modifications are made, the coast time in the teardrop from the injection point to the cusp is calculated:

$$t_{coast} = \tau - \frac{1}{2}T_p - t_f. \tag{272}$$

Since the value of the LROE $\beta$ increases linearly with time, the change in $\beta$ during $t_{coast}$, or $\Delta\beta$, can be calculated as

$$\Delta\beta = t_{coast}\omega. \tag{273}$$

This means that the injection point into the teardrop trajectory must be at

$$\beta_t = \beta_{cutoff} - \Delta\beta, \tag{274}$$

where $\beta_t$ is the entry angle to target and $\beta_{cutoff}$ is the angle associated with the

teardrop cusp, the last point along the trajectory where injection is allowed. This ensures, given $t_f$, that upon maneuver completion, the inspector satellite coasts for the appropriate amount of time to obtain the desired lighting condition once it reaches the top of the teardrop. This $\beta_t$, along with the other previously chosen LROEs, provide the target state, $X_t$, via Equations 76–81 to target for the end of the coast-burn-coast-burn sequence.

The other type of sunlight constraint, denoted the relaxed (or soft) sunlight constraint, allows an angular margin, $\theta_s$, from the tight sunlight constraint requirement. Thus, $\theta_s$ is chosen by the user, and the maximum and minimum values of $\tau$ are calculated by

$$\tau_{max} = \frac{\alpha_s + \theta_s}{\omega} \tag{275}$$

$$\tau_{min} = \frac{\alpha_s - \theta_s}{\omega}. \tag{276}$$

Checks 1, 2, and 4 from the tight sunlight constraint are thus modified, using $\tau_{max}$ in the if statement instead of $\tau$, and for any true statements, both $\tau$ and $\tau_{max}$ (and $\tau_{min}$) are increased by $P$. The target LROE $\beta_t$ is calculated the same way with $\tau$, and then the bounds on $\beta$ are defined as

$$\beta_u = \min(\beta_{cutoff}, \beta_t + \theta_s) \tag{277}$$

$$\beta_l = \beta_t - \theta_s, \tag{278}$$

where the upper bound on $\beta$ can't go beyond $\beta_{cutoff}$. Then, instead of a target state, a target trajectory is used where the satellite may enter the trajectory anywhere within the bounds on $\beta$.

It may also be desirable that no other bright celestial objects appear in the sensor's field of view for some time during a portion of the teardrop. Due to the nature of a

lower teardrop where $D$ is negative, the Earth will not appear in the field of view. However, there are times when the Moon may appear in the field of view, unless the maneuver is modified to ensure that there is some angular margin, $\theta_m$ or greater, which exists between the in-plane angle to the unit vector pointing from the inspector satellite to the RSO, $\alpha_{d2c}$, and the in-plane angle to the projected vector pointing from the RSO to the Moon, $\alpha_{c2m}$. This margin of $\theta_m$ or greater must exist for a specified fraction of $T_p$, denoted $t_m$, where the desired time for the field of view to be clear is $t_{clear} = t_m T_p$. Thus, $\alpha_{d2c}$ and $\alpha_{c2m}$ are calculated during $t_{clear}$, with an example shown in Figure 40. In a case like this, the maneuver must be modified such that there is no crossing of the two angles during $t_{clear}$, and also such that $\theta_m$ or more exists between the two angles throughout $t_{clear}$.

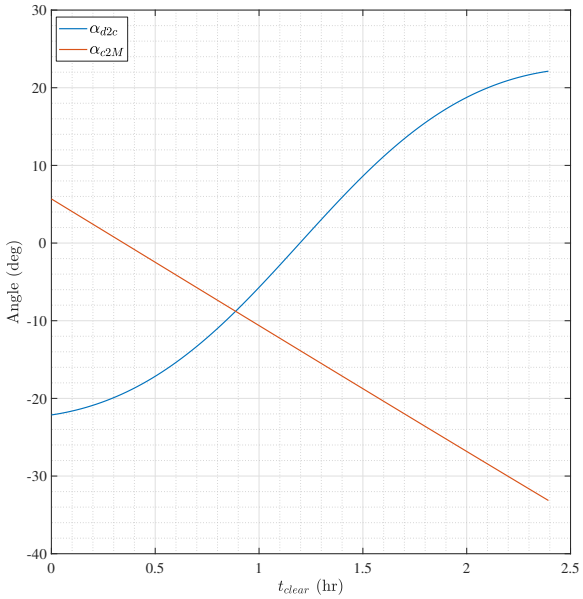



**Figure 40. Example Moon Conflict for Teardrop**

For both the tight and relaxed sunlight constraint, the first step is to determine the minimum angle between $\alpha_{d2c}$ and $\alpha_{c2m}$ during $t_{clear}$, or $\alpha_{min}$. If $\alpha_{min} < \theta_m$, then $\beta_t$ and thus $X_t$ must be shifted. The direction and amount of the shift is determined by examining $\alpha_{c2m}$ if the entry into the teardrop were both shifted backward or forward

along the trajectory. Once the minimum angle between the shifted $\alpha_{c2m}$ and $\alpha_{d2c}$ for $t_{clear}$ is greater than or equal to $\theta_m$, a new $\beta_t$ must be defined. If the tight sunlight constraint is enabled, then the new $\beta_t$ corresponding to the new $X_t$ is $\beta_t = \beta_t + t_{shift}\omega$. If the relaxed sunlight constraint is enabled, then for cases where the arrival is shifted forward in the trajectory, (but not entering past $\beta_{cutoff}$), then $\beta_l = \beta_t$, where $\beta_t$ is the new, shifted value. For cases where the arrival is shifted backward in the trajectory, then $\beta_u = \beta_t$.

For cases where the original $\alpha_{min} \geq \theta_m$ and the relaxed sunlight constraint is also enabled, there is some additional checking to do. $\beta_u$ and $\beta_l$ may need to be modified to prevent cases where the optimal entry angle within the bounds may create a Moon conflict scenario. Thus, if $\alpha_{min} \geq \theta_m$, and if $\alpha_{min} < \theta_m + \theta_s$, then the bounds on $\beta$ must be adjusted. If this is the case, then similar to before, $\alpha_{min}$ is recalculated by shifted forward or backward the entry into the teardrop until $\alpha_{min} \leq \theta_m$. If this inequality becomes true for larger values of $\beta$, then $\beta_u$ is decreased (as long as it is still less than $\beta_{cutoff}$), to $\beta_u = \beta_t + t_{shift}\omega$. If the inequality becomes true for lower values of $\beta$, then $\beta_l$ is increased to $\beta_l = \beta_t + t_{shift}\omega$, where $t_{shift}$ would be negative.

In summary, for the tight sunlight constraint, $X_t$ is determined in order to obtain the exact desired lighting condition. If the Moon avoidance constraint is also active, then $X_t$ may be shifted to avoid any conflicts. For the relaxed sunlight constraint, bounds on $\beta$ are determined in order to allow entry into the trajectory at any point within those bounds. If the Moon avoidance constraint is also active, then the bounds on $\beta$ are adjusted such that no Moon conflict occurs. Thus, sunlight and field-of-view constraints can be incorporated into the problem formulation without creating path constraints, where the entry point or bounds on the entry point are simply shifted. It should be noted, however, that these methods have been developed for Sun and Moon vectors projected into the orbital plane of the RSO. This approximation assumes that

the projections of these vectors are reasonably close to the actual three-dimensional vectors.

### 4.4.6 Collision Avoidance.

This subsection develops a method to avoid collisions between the inspector satellite and the RSO by adding inequality constraints to the optimization problem. To begin, it is assumed that the initial trajectory the spacecraft is following is a safe one, and will not collide with the RSO for a safe amount of time. Given this, there are then two types of collisions which must be avoided. The first collision which must be avoided is one which could occur after the first burn in the coast-burn-coast-burn sequence. This means that the transfer trajectory created by the first burn must avoid an exclusion zone which is modeled here by a keep-out ellipsoid centered around the RSO. The second type of collision which must be avoided is a passive collision, which is one which would occur in a certain amount of time (defined by the user) if the second (final) burn in the coast-burn-coast-burn sequence failed to occur. The method developed in this study to avoid both types of collisions is to use an analytic expression for the three final HCW positions after the first burn, propagated forward to $N$ discrete points in time, $t_i$, past the beginning of the second coast phase. Thus, there are $N$ inequality constraints contained in $g$:

$$g_i = 1 - \left( \frac{x_{c2_i}^2}{x_{ell}^2} + \frac{y_{c2_i}^2}{y_{ell}^2} + \frac{z_{c2_i}^2}{z_{ell}^2} \right) \leq 0, \qquad i = 1, ..., N, \tag{279}$$

where $x_{ell}$, $y_{ell}$, and $z_{ell}$ define the size of the keep-out ellipsoid which is centered about the non-maneuvering RSO at the origin of the relative frame. The discrete positions

along the second coast arc (and beyond for passive collision avoidance) are

$$x_{c2_i} = \frac{\dot{x}_{b1_f} \sin(t_i\omega)}{\omega} - \frac{\dot{y}_{b1_f} (2\cos(t_i\omega) - 2)}{\omega} - x_{b1_f} (3\cos(t_i\omega) - 4) \tag{280}$$

$$\begin{aligned} y_{c2_i} = {} & y_{b1_f} + x_{b1_f} (6\sin(t_i\omega) - 6t_i\omega) \\ & + \frac{\dot{y}_{b1_f} (4\sin(t_i\omega) - 3t_i\omega)}{\omega} + \frac{\dot{x}_{b1_f} (2\cos(t_i\omega) - 2)}{\omega} \end{aligned} \tag{281}$$

$$z_{c2_i} = z_{b1_f} \cos(t_i\omega) + \frac{\dot{z}_{b1_f} \sin(t_i\omega)}{\omega}, \tag{282}$$

where the initial conditions are $X_{b1_f}$, determined by Equations 261–266. Thus, the user prescribes the value for $N$, the number of times the trajectory after the first burn will be checked until a certain amount of time has passed after $t_f$, which time is also prescribed by the user. This amount of time checked after $t_f$ is a lower bound to the amount of time the passive collision avoidance is enforced. The higher $N$ is, the more dense the checks are and the more inequality constraints exist to ensure the spacecraft does not enter the keep-out zone. The $N$ inequality constraints are analytic constraint functions, and can be evaluated very quickly inside of an optimization routine, as well as be used to calculate analytic derivatives. As long as the sampling density is sufficiently high, this method ensures that the inspector satellite does not enter the keep-out ellipsoid after the first burn or if the second burn fails to occur.

### 4.4.7 Optimization Problem Formulations.

In order to determine the minimum fuel or the minimum time required to maneuver into a prescribed relative teardrop trajectory, multiple optimization problems have been formulated depending on the type of sunlight constraint enabled. The resulting optimization problems are all nonlinear, non-convex, constrained optimization problems which require a good initial guess in order to converge to the desired local minimum. Due to its established performance on other subproblems, MATLAB's GA

is used to generate an initial guess. The solution from the GA is then used as the initial guess for MATLAB's *fmincon*, which may further refine the solution and take advantage of gradient information.

#### 4.4.7.1 Minimum-Fuel Formulation.

The goal of the minimum-fuel formulation is to minimize the engine-on time, $t_{on} = d_{b1} + d_{b2}$, for a given fixed final time, $t_f$. Since $t_f$ is a constant, the cost function can be expressed as:

$$J = t_{2_f}(t_{c_f} - t_{1_f} t_{c_f} - 1), \tag{283}$$

where the optimization variables, their lower and upper bounds, and any equality or inequality constraints depend on whether the tight or relaxed sunlight constraint is enabled.

**4.4.7.1.1 Tight Sunlight Constraint.** If the tight sunlight constraint is enabled, then the optimization variables are

$$\chi = [t_{1_f}, \alpha_1, \phi_1, t_{c_f}, \alpha_2, \phi_2, t_{2_f}], \tag{284}$$

with lower and upper bounds

$$\chi_l = \left[0, 0, -\frac{\pi}{2}, 0, 0, -\frac{\pi}{2}, 0\right] \tag{285}$$

$$\chi_u = \left[1, 2\pi, \frac{\pi}{2}, 1, 2\pi, \frac{\pi}{2}, 1\right], \tag{286}$$

and the equality constraints are

$$h = X_f - X_t = 0. \tag{287}$$

If collision avoidance is enabled, then the problem is also subject to the $N$ inequality constraints as shown in Equation 279.

**4.4.7.1.2 Relaxed Sunlight Constraint.** If the relaxed sunlight constraint is enabled, then there is one additional optimization variable,

$$\chi = [t_{1_f}, \alpha_1, \phi_1, t_{c_f}, \alpha_2, \phi_2, t_{2_f}, \beta], \tag{288}$$

with lower and upper bounds

$$\chi_l = \left[0, 0, -\frac{\pi}{2}, 0, 0, -\frac{\pi}{2}, 0, \beta_l\right] \tag{289}$$

$$\chi_u = \left[1, 2\pi, \frac{\pi}{2}, 1, 2\pi, \frac{\pi}{2}, 1, \beta_u\right], \tag{290}$$

and the equality constraints are

$$h = \begin{bmatrix} x_f \\ y_f \\ z_f \\ \dot{x}_f \\ \dot{y}_f \\ \dot{z}_f \end{bmatrix} - \begin{bmatrix} \frac{-a_e}{2}\cos\beta + x_d \\ a_e \sin\beta + y_d \\ z_{max}\sin(\gamma+\beta) \\ \frac{a_e}{2}\omega\sin\beta \\ a_e\omega\cos\beta - \frac{3}{2}\omega x_d \\ z_{max}\omega\cos(\gamma+\beta) \end{bmatrix} = 0, \tag{291}$$

where $\beta$ is now allowed to vary within its bounds to further minimize the cost function. If collision avoidance is enabled, then the problem is also subject to the $N$ inequality

constraints as shown in Equation 279.

#### 4.4.7.2 Minimum-Time Formulation.

The goal of the minimum-time formulation is to minimize the total time of the maneuver, or $t_f$, where $t_f$ becomes one of the optimization variables for this type of problem. The cost function is thus:

$$J = t_f, \tag{292}$$

where the optimization variables are

$$\chi = [t_f, t_{1_f}, \alpha_1, \phi_1, t_{c_f}, \alpha_2, \phi_2, t_{2_f}, \beta], \tag{293}$$

and where their lower and upper bounds, and any equality or inequality constraints depend on whether the tight or relaxed sunlight constraint is enabled.

**4.4.7.2.1 Tight Sunlight Constraint.** If the tight sunlight constraint is enabled, then bounds on the optimization variables are

$$\chi_l = \left[0, 0, 0, -\frac{\pi}{2}, 0, 0, -\frac{\pi}{2}, 0, \beta_{MIN}\right] \tag{294}$$

$$\chi_u = \left[t_{f_{max}}, 1, 2\pi, \frac{\pi}{2}, 1, 2\pi, \frac{\pi}{2}, 1, \beta_{cutoff}\right], \tag{295}$$

where $t_{f_{max}}$ should be chosen to be large enough to allow a solution to be found. The bounds on $\beta$ allow the entry angle to vary from the point of maximum coast time up until the last point allowed for entry, $\beta_{cutoff}$. The $\beta$ angle associated with the point of maximum coast time is found by setting $t_f$ equal to zero, and calculating the

maximum possible coast time,

$$t_{coast_{max}} = \tau - \frac{1}{2}T_p, \tag{296}$$

and thus

$$\beta_{MIN} = \beta_{cutoff} - t_{coast_{max}}\omega. \tag{297}$$

The equality constraints are the same as Equation 291 with one additional equality constraint, $h_7$,

$$h_7 = \tau - \frac{1}{2}T_p - t_f - \frac{(\beta_{cutoff} - \beta)}{\omega} = 0, \tag{298}$$

which ensures that the tight sunlight constraint is met. If collision avoidance is enabled, then the problem is also subject to the $N$ inequality constraints as shown in Equation 279.

**4.4.7.2.2 Relaxed Sunlight Constraint.** If the relaxed sunlight constraint is enabled, then the lower bound on $\beta$ is modified to:

$$\beta_{MIN} = \beta_{cutoff} - t_{coast_{max}}\omega = \beta_{cutoff} - (\tau_{max} - \frac{1}{2}T_p)\omega. \tag{299}$$

The equality constraints are the same as Equation 291, and there are now two inequality constraints described by,

$$g = \begin{bmatrix} \tau_{min} - \frac{1}{2}T_p - t_f - \frac{\beta_{cutoff} - \beta}{\omega} \\ t_f + \frac{\beta_{cutoff} - \beta}{\omega} - \tau_{max} + \frac{1}{2}T_p \end{bmatrix} \leq 0, \tag{300}$$

where this ensures that the inspector satellite enters the trajectory at a $t_f$ where it will satisfy the relaxed sunlight constraint. If collision avoidance is enabled, then the problem is also subject to the additional $N$ inequality constraints as shown in Equation 279.

#### 4.4.7.3 Analytic Derivatives.

Since all of the cost functions and constraint functions have analytic expressions as functions of the optimization variables, the analytic gradients can be found and supplied to the NLP solver. Thus, the analytic gradients of the objective functions and the analytic Jacobians of the constraint functions have been developed by using MATLAB's Symbolic Toolbox for the minimum-fuel problems. Specifically, to produce the analytic gradient of the objective functions, the following line of code is executed: $gradient(J, \chi)$. To produce the analytic Jacobians of the constraint functions, the following line of code is executed: $jacobian(h, \chi)$. The analytic gradients of the objective functions, $\nabla J$, are fairly simple, whereas the Jacobian of the equality and inequality constraints, $\nabla h$ and $\nabla g$, are quite complex and far too lengthy to present. The Hessian has not been calculated analytically (although it could be) and thus is approximated by the NLP solver.

### 4.4.8 Simulations and Results.

This subsection presents multiple simulation results in order to demonstrate the efficacy of the developed methods. Three parameter sets are used to produce four sets of results. The first set of results illustrates the effects of the tight and relaxed sunlight constraints, and compares the new coast-burn-coast-burn sequence to the burn-coast-burn sequence. The next set of results demonstrates the effects of the Moon avoidance option, for both the tight and relaxed sunlight constraints. The following set of results

shows both active and passive collision avoidance scenarios, where the developed collision avoidance methodology ensures a safe maneuver. Finally, the last set of results presents a range of optimal solutions for a specific scenario, showing the Pareto front of optimal solutions and their corresponding trajectories.

#### 4.4.8.1 Sunlight Constraints.

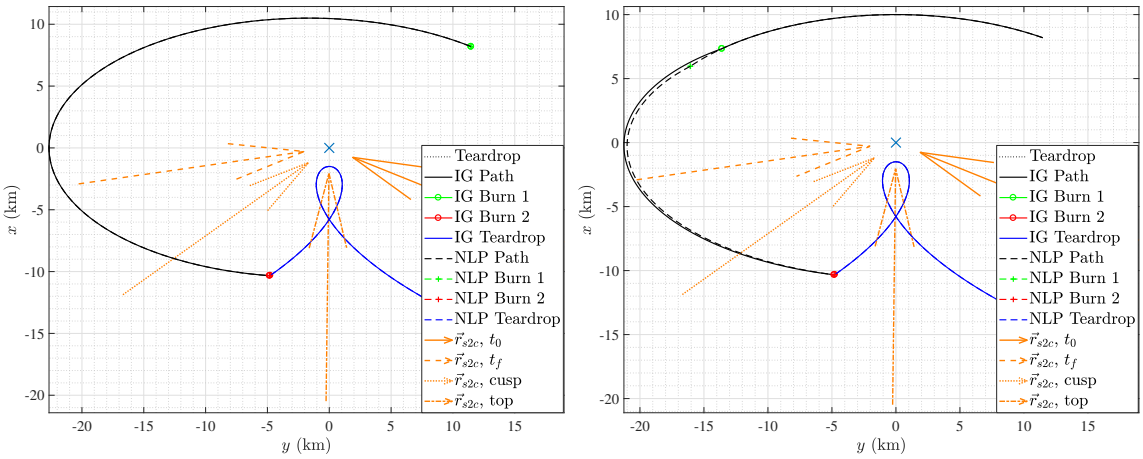
The first results presented use the parameter set A shown in Table 17, where the time is given in UTC. The semi-major axis of the RSO is represented by $a_c$, and $f_0$ is the initial true anomaly of the RSO, measured from the ECI $\hat{I}$ axis, which along with $a_c$ defines the initial inertial state of the RSO since the RSO is assumed to be in a circular, zero-inclination orbit. The initial conditions of the inspector satellite are the initial states in the relative frame and for this scenario correspond to the satellite starting out in an NMC with $a_e = 20$ km, $y_{d_0} = 0$ km, $z_{max} = 2$ km, $\gamma = 0^o$, and $\beta_0 = 145^o$.

**Table 17. Problem B-3 Simulation Parameters, Set A**

| Sun and Time | RSO Properties | Inspector Properties | Teardrop Parameters |
|---|---|---|---|
| $\theta_s = 45^o$ | $a_c = 42,164.137$ km | $a_0 = 0.02$ N/kg | $D = -1.5$ km |
| $\text{Year}_0 = 2017$ | $f_0 = 45^o$ | $c = 3.33$ km/s | $T_p = 0.3P$ |
| $\text{Month}_0 = \text{Aug}$ | | $x_0 = 8.192$ km | $y_T = 0$ km |
| $\text{Day}_0 = 28$ | | $y_0 = 11.472$ km | $z_{max} = 2$ km |
| $\text{Hr}_0 = 4$ | | $z_0 = 1.147$ km | $\gamma = 0^o$ |
| $\text{Min}_0 = 0$ | | $\dot{x}_0 = 0.4183$ m/s | |
| $\text{Sec}_0 = 0$ | | $\dot{y}_0 = -1.1947$ m/s | |
| | | $\dot{z}_0 = -0.1195$ m/s | |

The GA is used to produce an initial guess for these minimum-fuel solutions, and MATLAB's *fmincon* is used as the NLP solver to further refine the solutions from the GA. The tight sunlight constraint is enabled first, with results shown in Figure 41, where $t_f$ was chosen to be 14 hours. Figure 41 (a) shows the burn-coast-burn solution and has an engine-on time, or $t_{on}$, of $t_{on} = 68.39$ seconds. Figure 41 (b) shows the modified trajectory when a coast phase is allowed before the burn-coast-

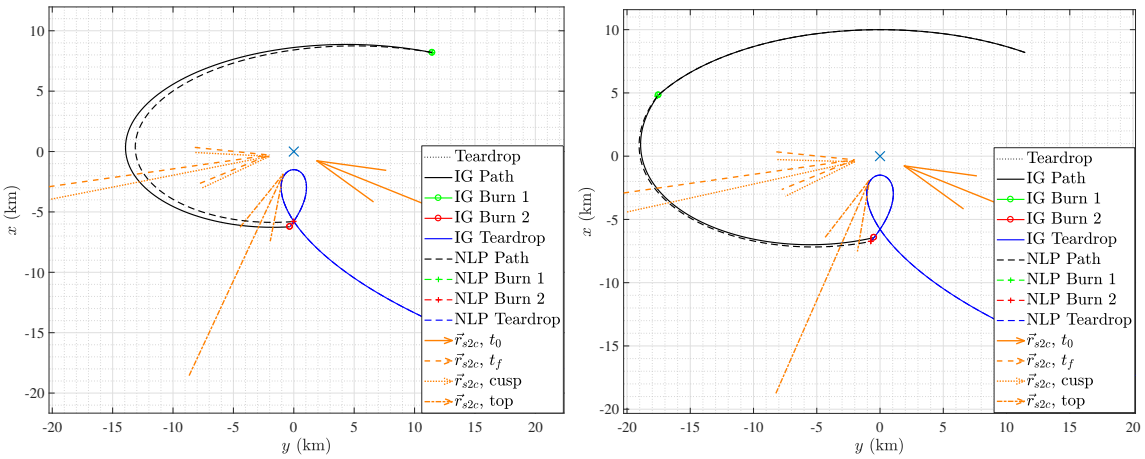
burn sequence, with a reduced engine-on time of $t_{on} = 62.01$ seconds. Thus it can be seen that for this scenario, the additional coast phase in front of the burn-coast-burn sequence saves fuel compared to the previously developed burn-coast-burn sequence. Note that for both scenarios the NLP solver solution falls almost directly on top of the GA solution, showing that the initial guess method provides a good initial guess (denoted IG in the plots). The effects of the tight sunlight constraint can be seen in both of these plots, where the projected vector from the Sun to the RSO, $r_{s2c}$, is shown at four different times: at the beginning of the maneuver, $t_0$; at the end of the maneuver, $t_f$; at the time that the inspector satellite reaches the teardrop cusp; and, at the time the inspector satellite reaches the top of the teardrop. In both plots, it can be seen that the inspector satellite successfully maneuvers into the teardrop such that the projected sunlight vector aligns with the line-of-sight vector from the inspector satellite to the RSO once the inspector satellite reaches the top of the teardrop.




(a) Burn-Coast-Burn (b) Coast-Burn-Coast-Burn

**Figure 41. B-3, Tight Sunlight Constraint, Min Fuel Solution, $t_f = 14$ hrs**

Next, the relaxed sunlight constraint is enabled, allowing the inspector satellite to enter the trajectory within determined bounds such that the projected sunlight vector is within $\pm\theta_s$ of the tight sunlight constraint. The results can be seen in Figure 42, where (a) shows the result with just the burn-coast-burn sequence, and




(a) Burn-Coast-Burn (b) Coast-Burn-Coast-Burn

**Figure 42. B-3, Relaxed Sunlight Constraint, Min Fuel Solution, $t_f = 14$ hrs**

(b) shows it with the coast-burn-coast-burn sequence. Without the additional coast phase, $t_{on} = 59.61$ seconds, whereas with the additional coast phase, $t_{on} = 47.85$ seconds. Thus, as expected with the relaxed sunlight constraint, both cases have lower engine-on times than their respective tight sunlight constraint cases, and there is a 19.73% fuel savings for the relaxed sunlight constraint scenario if there is an additional coast phase included before the burn-coast-burn sequence.

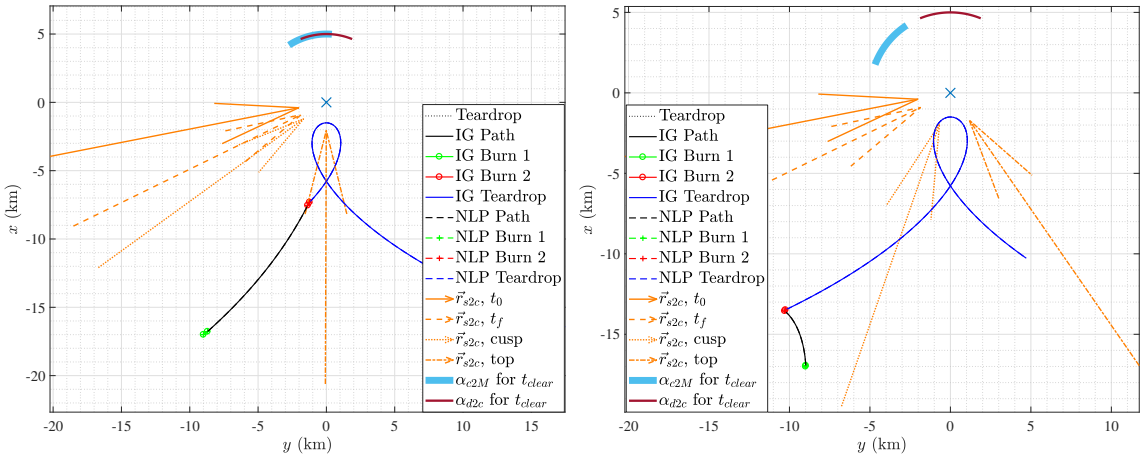
#### 4.4.8.2 Moon Avoidance Constraint.

Now, a scenario is presented where there is a Moon conflict, i.e., the in-plane vector from the RSO to the Moon comes within $\theta_m$ of the in-plane vector from the inspector satellite to the RSO (thus it is in the sensor's field of view) during the desired time for no conflicts to occur, $t_{clear}$. Table 18 shows the parameter set B used for this scenario, where the parameters were chosen to produce a Moon conflict.

The tight sunlight constraint is enabled first with results seen in Figure 43, where the fixed final time for these cases is $t_f = 1$ hour. Figure 43 (a) shows the Moon conflict of the resulting trajectory, which was the example for Figure 40, showing there is a conflict between the two in-plane angles during the prescribed period of

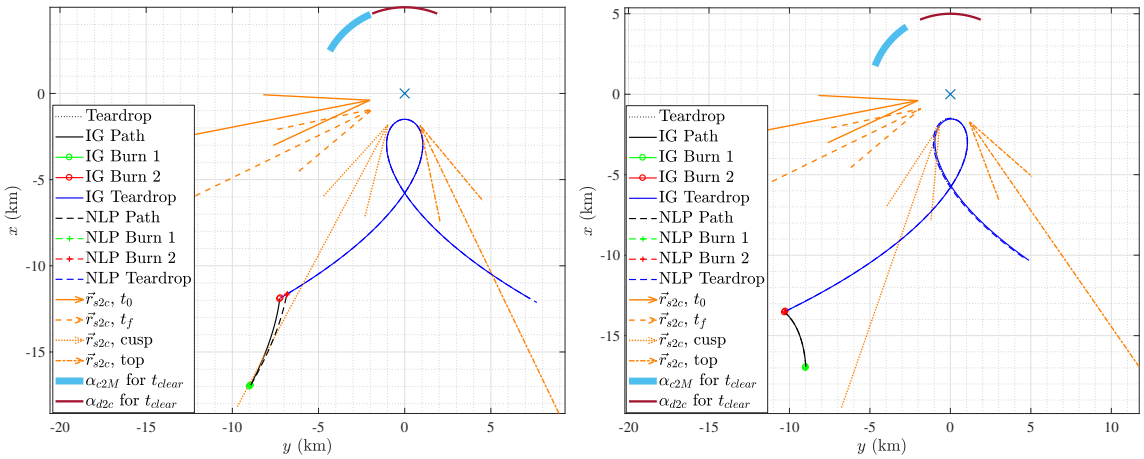
**Table 18. Problem B-3 Simulation Parameters, Set B**

| Sun and Time | RSO Properties | Inspector Properties | Teardrop Parameters |
|---|---|---|---|
| $\theta_s = 45^o$<br>$\text{Year}_0 = 2017$<br>$\text{Month}_0 = \text{Sept}$<br>$\text{Day}_0 = 4$<br>$\text{Hr}_0 = 23$<br>$\text{Min}_0 = 0$<br>$\text{Sec}_0 = 0$ | $a_c = 42,164.137$ km<br>$f_0 = 265^o$ | $a_0 = 0.02$ N/kg<br>$c = 3.33$ km/s<br>$x_0 = -17$ km<br>$y_0 = -9$ km<br>$z_0 = 0$ km<br>$\dot{x}_0 = 0$ m/s<br>$\dot{y}_0 = 0.5$ m/s<br>$\dot{z}_0 = 0$ m/s | $D = -1.5$ km<br>$T_p = 0.3P$<br>$y_T = 0$ km<br>$z_{max} = 0$ km<br>$\gamma$ =NA |




(a) Moon Conflict (b) Moon Conflict Avoided

**Figure 43. B-3, Moon Avoidance, Tight Sunlight Constraint, Min Fuel, $t_f = 1$ hr**




(a) Moon Conflict (b) Moon Conflict Avoided

**Figure 44. B-3, Moon Avoidance, Relaxed Sunlight Constraint, Min Fuel, $t_f = 1$ hr**

time where no conflicts are desired, where $t_{clear} = \frac{1}{3}T_p$ for this case. The engine-on time for this case is $t_{on} = 300.31$ seconds. Figure 43 (b) then shows the result when the Moon avoidance constraint is enabled, where the point of entry into the teardrop trajectory has been shifted such that the two angles maintain $\theta_m = 10^o$ between them during $t_{clear}$. This case has an engine-on time of $t_{on} = 188.60$ seconds and has a much lower cost, which may be the case often, since the Moon avoidance constraint has forced the tight sunlight constraint to be violated. These results show how the developed methods successfully shift the entry into the trajectory such that there is no conflict.

With the relaxed sunlight constraint enabled, the results can be seen in Figure 44 where the fixed final time is again $t_f = 1$ hour. Figure 44 (a) shows the resulting Moon conflict with an engine-on time of $t_{on} = 149.20$ seconds. Note that since the relaxed sunlight constraint was enabled, under certain scenarios the solution may avoid a Moon conflict without enabling the Moon avoidance constraint. However, with the Moon avoidance constraint enabled, it is ensured that the solution avoids the Moon conflict and maintains at least $\theta_m$ during $t_{clear}$, as seen in Figure 44 (b). This solution appears to be the same as the tight sunlight constraint scenario, with an engine-on time of $t_{on} = 188.60$ seconds, since the shifted $\beta_t$ value associated with the shifted $X_t$ from the tight sunlight constraint scenario was set as the upper bound on $\beta$ for the relaxed sunlight constraint case.

#### 4.4.8.3 Collision Avoidance.

The developed collision avoidance constraints can be enabled in any one of the cases already presented. The first collision avoidance example presented here demonstrates how if the inspector satellite is going to enter the defined exclusion zone during the second coast period (and before the second burn), then the burns will be modified

such that the resulting trajectory avoids the exclusion zone. The parameters used for this simulation are parameter set C shown in Table 19, where the inspector satellite's initial conditions come from initially being in an NMC with $a_e = 5$ km, $y_{d_0} = 0$ km, $z_{max} = 2$ km, $\gamma = 180^o$, and $\beta_0 = 45^o$. The collision avoidance parameters are defined in Table 20.

**Table 19. Problem B-3 Simulation Parameters, Set C**

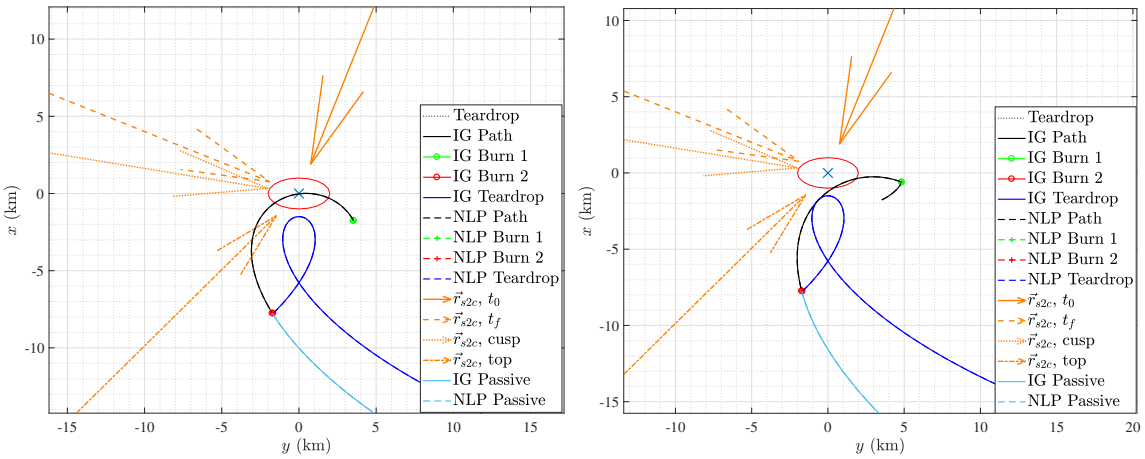
| Sun and Time | RSO Properties | Inspector Properties | Teardrop Parameters |
|---|---|---|---|
| $\theta_s = 45^o$ | $a_c = 42,164.137$ km | $a_0 = 0.02$ N/kg | $D = -1.5$ km |
| $\text{Year}_0 = 2017$ | $f_0 = 135^o$ | $c = 3.33$ km/s | $T_p = 0.3P$ |
| $\text{Month}_0 = \text{Aug}$ | | $x_0 = -1.7678$ km | $y_T = 0$ km |
| $\text{Day}_0 = 28$ | | $y_0 = 3.5355$ km | $z_{max} = 2$ km |
| $\text{Hr}_0 = 4$ | | $z_0 = -1.4142$ km | $\gamma = 0^o$ |
| $\text{Min}_0 = 0$ | | $\dot{x}_0 = 0.1289$ m/s | |
| $\text{Sec}_0 = 0$ | | $\dot{y}_0 = 0.2578$ m/s | |
| | | $\dot{z}_0 = -0.1031$ m/s | |

**Table 20. Collision Avoidance Parameters**

| $x_{ell}$ (km) | $y_{ell}$ (km) | $z_{ell}$ (km) | Passive Time (Lower Bound) | $N$ |
|---|---|---|---|---|
| 1 | 2 | 1 | $\geq 0.55t_f$ | 80 |

The results can be seen in Figure 45 for a fixed final time of $t_f = 6$ hours and with the relaxed sunlight constraint enabled. Figure 45 (a) shows how, if left unmodified, the trajectory enters the exclusion zone (represented by the red ellipse) on its way to the injection point. Figure 45 (b) shows the exact same scenario, but with the collision avoidance constraint enabled, demonstrating how the inspector satellite coasts at the beginning, followed by the first burn which leads to a trajectory that avoids the exclusion zone and still satisfies the relaxed sunlight constraint. This 2-D plot may appear as if the exclusion zone is still slightly violated, but in 3-D in Figure 46 it can be seen that the trajectory actually travels underneath the $xy$ plane, and just touches the edge of the exclusion zone, with $N = 80$.

The second collision avoidance example presented here demonstrates the passive collision avoidance capability. Thus, a scenario has been developed where, if the




(a) Collision Scenario (b) Collision Avoided

**Figure 45. B-3, Active Collision Avoidance, Relaxed Sunlight, Min Fuel, $t_f = 6$ hr**

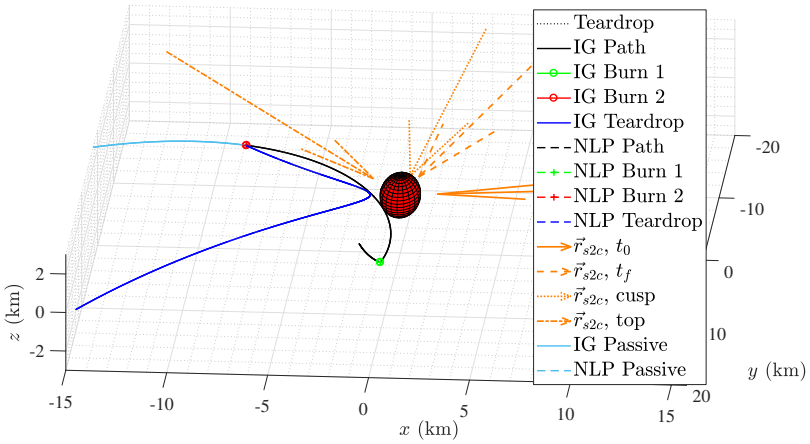
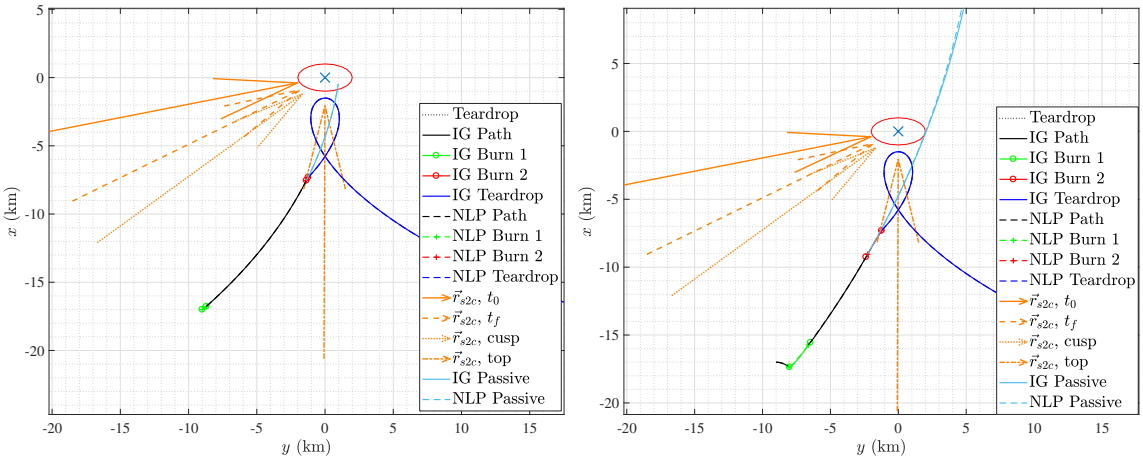



**Figure 46. B-3, Active Collision Avoided, Relaxed Sunlight, Min Fuel, $t_f = 6$ hr, 3-D**

second burn fails to take place, the inspector satellite would continue coasting into the exclusion zone. For this scenario, parameter set B from Table 18 is used, along with the same collision avoidance parameters in Table 20, and the tight sunlight constraint is enabled with a fixed final time of $t_f = 1$ hour. Figure 47 (a) shows the scenario without the collision avoidance constraint enabled, where it can be seen that if the satellite fails to perform the second burn, then the satellite would coast into the exclusion zone. Figure 47 (b) shows the exact same scenario again, but now with the collision avoidance constraint enabled. Thus the maneuver has been successfully

modified, coasting for a period of time before making longer burns such that it is passively safe, but still arrives at $X_t$ in the given amount of time, $t_f$.




(a) Passively Unsafe (b) Passively Safe

**Figure 47. B-3, Passive Collision Avoidance, Tight Sunlight, Min Fuel, $t_f = 1$ hr**

#### 4.4.8.4 Summary of Simulation Data.

Table 21 shows the summary of scenarios presented, where each row contains the following: the parameter set used, whether the new coast-burn-coast-burn sequence was used or not, which constraints were active, the final time used, the algorithm used inside *fmincon*, the engine-on time or the total burn time from both the GA and the NLP solver, and the computation time from both the GA and the NLP solver. Regarding the algorithm used within *fmincon*, the *sqp* solution was used unless it did not converge to an exit flag of 1 (meaning a local minimum was found), in which case the solution from the *interior-point* method was used. For all scenarios, first derivative information was supplied to the solver and the solver successfully converged to a local minimum. All other solver settings were left at their default values, with the exception of increasing the maximum number of iterations and function evaluations, and tightening the step size tolerance to $1 \times 10^{-16}$. One important takeaway from Table 21 are the computation times. Due to the analytic propagation of the coast-

burn-coast-burn sequence and the way in which the constraints were formulated, the computation times are fast, on the order of 10 seconds for the GA and on the order of 1 second or less for the NLP solver. The only exception is when the collision avoidance method is enabled, which increases the computation time of the GA to about 1.5 minutes.

**Table 21. Problem B-3 Simulation Results**

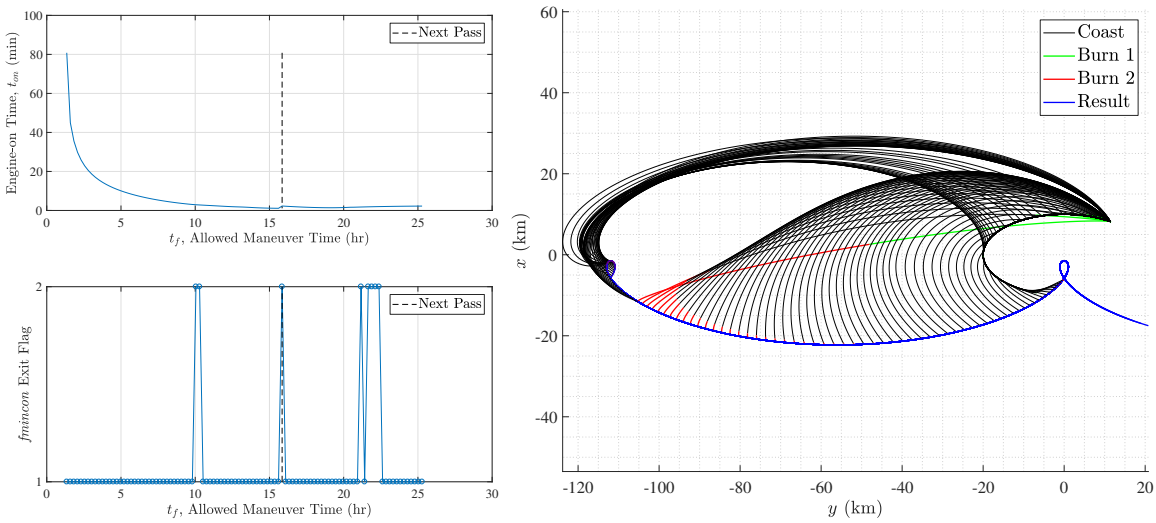
| Param Set | First Coast | Constraints | | | $t_f$ (hr) | *fmincon* algorithm | Burn Time (s) | | CPU Time (s) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Sun | Moon | Collision | | | GA | NLP | GA | NLP |
| A | off | tight | off | off | 14 | *sqp* | 68.39 | 68.39 | 7.52 | 0.14 |
| A | on | tight | off | off | 14 | *int-pt* | 63.97 | 62.01 | 9.42 | 0.83 |
| A | off | relaxed | off | off | 14 | *sqp* | 59.61 | 59.61 | 11.42 | 0.01 |
| A | on | relaxed | off | off | 14 | *int-pt* | 50.44 | 47.85 | 10.65 | 2.69 |
| B | on | tight | off | off | 1 | *int-pt* | 303.3 | 300.3 | 5.53 | 0.53 |
| B | on | tight | on | off | 1 | *int-pt* | 189.4 | 188.6 | 6.87 | 0.77 |
| B | on | relaxed | off | off | 1 | *int-pt* | 151.3 | 149.2 | 8.25 | 0.81 |
| B | on | relaxed | on | off | 1 | *int-pt* | 188.9 | 188.6 | 8.01 | 4.61 |
| B | on | tight | off | on | 1 | *sqp* | 1252 | 867.9 | 94.1 | 1.93 |
| C | on | relaxed | off | off | 6 | *sqp* | 118.1 | 118.1 | 8.60 | 0.03 |
| C | on | relaxed | off | on | 6 | *sqp* | 143.1 | 141.7 | 101.0 | 11.3 |

#### 4.4.8.5 Range of Optimal Solutions.

If the fixed final time of the maneuver, $t_f$, does not have to be a specific value, then a range of optimal solutions can be generated for varying fixed final times. The strategy employed to generate such a plot is to first solve the minimum-time problem for the given scenario. Then, for a range of fixed final times prescribed by the user, the minimum-fuel solution is calculated for the discrete fixed final times. The GA is used at the beginning, to help find an initial guess for the minimum-time problem, and then the solution is fed to *fmincon* for refinement. Then, for the next fixed final time, the *fmincon* solution from the previous fixed final time is used as the initial guess. If the fixed final time reaches the point where the inspector satellite must wait until the next sunlight opportunity, then the GA is used again to find an initial guess for the first solution corresponding to the next opportunity. Once the range of

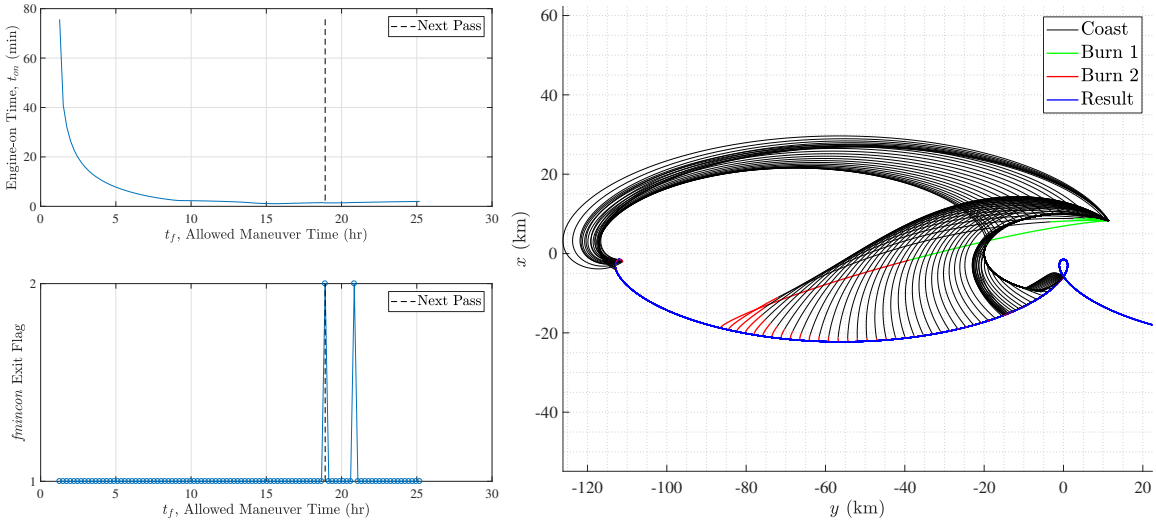
optimal solutions is generated, the user can examine the Pareto front of solutions and determine the best solution, weighing engine-on time, $t_{on}$, vs. the amount of time required to maneuver, $t_f$.

Figure 48 shows an example scenario with the parameters from set A in Table 17 and the tight sunlight constraint enabled, where the range of fixed final times are discretized from the minimum-time solution, $t_{f_{min}} = 1.347$ hours, to $t_{f_{min}}$ plus one period of the RSO. The minimum-fuel solution is calculated one-hundred times with the *sqp* algorithm used inside *fmincon*. These solutions take a total of 11.771 seconds to compute, where one minimum-time GA, one minimum-fuel GA, and one-hundred minimum-fuel *fmincon* solutions were generated in that amount of time. Figure 48 (a), subplot 1 shows the Pareto front of optimal solutions, with the minimum-time solution being the one at the very left. At $t_f = 15.85$ hours, the maneuver time is too long to enter the teardrop and meet the tight sunlight constraint, thus that solution and all other solutions to the right are calculated for the next opportunity to meet the tight sunlight constraint. Figure 48 (a) subplot 2 shows the *fmincon* exit flags corresponding to each solution. An exit flag of 1 means that the first-order optimality measure was less than the tolerance and the maximum constraint violation was less than its tolerance, whereas an exit flag of 2 means that the change in the optimization variables was less than its tolerance and the maximum constraint violation was less than its tolerance. Thus, an exit flag of 1 is preferred, but an exit flag of 2 may also be acceptable, and still means a local minimum is possible. Figure 48 (b) shows the resulting trajectories, where the minimum-time solution is apparent by the engine being on the entire time. It can be seen that as $t_f$ increases, the inspector satellite continues coasting in its initial NMC until the point where it must make its first burn. Once $t_f$ increases to the point where it must wait for the next opportunity, the trajectories leave the initial NMC and head upwards and to the far left.




(a) $t_{on}$ and Exit Flags (b) Resulting Trajectories

**Figure 48. B-3, Range of Solutions, *fmincon sqp*, Tight Sunlight Constraint, Min Fuel**




(a) $t_{on}$ and Exit Flags (b) Resulting Trajectories

**Figure 49. B-3, Range of Solutions, *fmincon sqp*, Soft Sunlight Constraint, Min Fuel**

If the exact same scenario is run as before, but now with the relaxed sunlight constraint enabled, the results are as expected, shown in Figure 49. This range of solutions takes 18.05 seconds to compute. Figure 49 (b) shows how the solutions start grouping together close to $\beta_{cutoff}$, since the relaxed sunlight constraint allows more solutions to make the first opportunity, and less solutions have to wait for the next

opportunity, as can be seen in Figure 49 (a), subplot 1, where the opportunity divide is now at $t_f = 18.91$ hours.

### 4.4.9 Solution Validation.

The guidance solutions generated herein for each problem can be validated to a certain degree by using the control generated in an independent simulation with a corresponding high-accuracy propagator. The resulting trajectory can then be analyzed and compared against the desired trajectory to see if the results satisfy the mission planner. Thus, for many of the results throughout Problem B, FreeFlyer* has been used to visualize and validate the results. One example of some validation performed is shown in this subsection, for the passive collision avoidance scenario, where the collision avoidance constraint was enabled. Thus, the trajectory from Figure 47 (b) will be compared to the trajectory generated from FreeFlyer.

The FreeFlyer scenario is initiated with the parameters in Table 18. Thus, the initial date and time, the initial orbit of the RSO, and the initial relative state of the inspector satellite were set as the initial conditions in FreeFlyer. The engine properties of the inspector satellite were also set to produce the initial thrust-to-mass ratio, $a_0$, as well as the engine efficiency, $c$. The propulsion system is modeled by a chemical-spherical tank with a blow-down pressure model and thus continuous mass loss is accounted for. The integrator for both the RSO and the inspector satellite was the *Runge Kutta 8(9)* integrator with a fixed step size of 40 seconds and a relative error tolerance of $1 \times 10^{-9}$. The aerodynamic properties for both satellites were left as the default values, and aerodynamic and solar radiation pressure forces were accounted for. For both satellites, the Earth forces field type accounted for four zonal and four tesseral terms, and the forces from the Sun and the Moon were also taken

*FreeFlyer, (2017). *Engineer Version 7.2.1.* Lanham: a.i. solutions, Inc. `https://ai-solutions.com/freeflyer/`

into account.

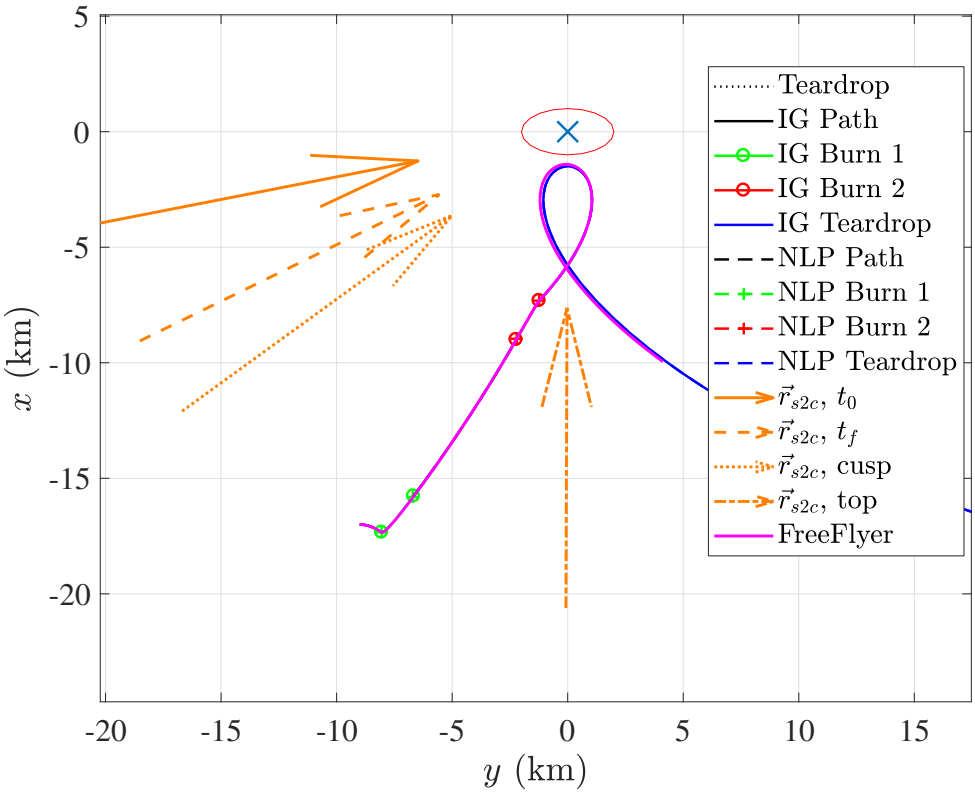



**Figure 50. FreeFlyer Validation Example**

The resulting trajectory can be seen in Figure 50, where the magenta curve shows the FreeFlyer trajectory. For this case, the trajectory resulting from the developed algorithms and the FreeFlyer trajectory are very similar, where the two teardrops are just slightly distinguishable. Rigorous comparisons could be performed on the results, depending on the level of analysis required in order to approve the generated guidance. Of course, these are just the open-loop trajectories, and don't include any feedback or closed-loop control, which would probably be incorporated into the expected trajectory as well. In summary, using an independent simulation with a high-accuracy propagator such as FreeFlyer can help to both visualize and validate the generated guidance, and show that the control profile generated by the developed algorithms works as intended and would be suitable for use on an actual satellite.

### 4.4.10 Problem B-3 Conclusion.

Both minimum-time and minimum-fuel optimization problems have been formulated and solved to find optimal maneuvers for an inspector satellite to enter a teardrop trajectory relative to an RSO in GEO, subject to lighting and collision constraints. A coast-burn-coast-burn sequence was developed and shown, in certain scenarios, that it can save fuel in a minimum-fuel problem compared to just a burn-coast-burn sequence. It was also demonstrated how for the non-periodic, relative teardrop trajectory, sunlight constraints can be formulated such that the inspector satellite obtains the desired lighting condition at the first feasible opportunity. The developed tight sunlight constraint successfully aligns the projected sunlight vector with the projected vector from the inspector satellite to the RSO at the top of the teardrop, where the relaxed sunlight constraint successfully allows an angular margin and further reduces the performance index, as desired. The Moon avoidance methods can also be enabled to ensure the Moon is not in the sensor's field of view during a prescribed portion of the teardrop, where the bounds of allowable entry into the teardrop are simply adjusted. The collision avoidance methods developed can also be enabled for fail-safe maneuvering, where both active and passive collision avoidance were demonstrated and the maneuvers were successfully modified to avoid entering a keep-out zone. It was also shown that a range of optimal solutions can be generated, allowing a mission planner to choose the best time vs. fuel solution from the Pareto front of optimal solutions. The generated control can also be simulated and validated with an independent, high-fidelity propagator, where one example showed that the generated guidance was suitable for use in a higher-fidelity environment. For all cases presented, the computation times were shown to be fast, making these algorithms suitable for rapid mission planning.

# V. Problem C

## 5.1 Overview

Unlike Problems A and B, Problem C assumes that the RSO maneuvers away from the reference orbit, and more specifically that the RSO maneuvers away optimally from the inspector satellite to evade the inspector satellite's inspection goal. Therefore, instead of formulating and solving one-sided optimal control problems as in Problems A and B, Problem C addresses pursuit-evasion games, or differential games, where the inspector satellite is denoted the pursuer and the maneuvering RSO is denoted the evader. The solutions from solving the games are open-loop strategies for the pursuer to achieve an inspection goal as soon as possible and for the evader to prolong it as long as possible, where the guidance for each player is generated from the known initial conditions. The games addressed are assumed to be urgent, where each player's thruster is continually on and using the maximum available thrust throughout the game, and the direction of the thrust vector is each player's control. This may be the case for two satellites with electric engines, where each satellite has one, body-fixed electric engine that is continually on during the game. Problem C also differs in that the Tschauner-Hempel equations of motion are used instead of the Hill-Clohessy-Wiltshire equations of motion, to allow the maneuvering RSO to have already departed its circular orbit. Metaheuristic methods will be used to solve multiple games, where each game considers a specific inspection goal. The inspection goals considered in this problem are the following: intercept, rendezvous, obtain Sun vector, match energy, obtain sun vector and match energy, and match energy and remain close.

This chapter is organized as follows. First, the equations of the motion and the control definition for both players are outlined. Next, the solution technique to be

used, the indirect heuristic method, and how it applies to the examined games is presented. The different game conditions, or inspection goals, are then developed, which are the terminal constraints for each game. Then, the techniques used to solve each game, again following the indirect heuristic method, are put forth. Finally, the results are presented, along with some validation efforts to give confidence that differential game solutions have indeed been found.

## 5.2 Equations of Motion and Control Definition

The equations of motion used for this problem are a heterogeneous form of the Tschauner-Hempel equations of motion, Equations 11–13, where the acceleration terms due to a constant, steerable thrust are retained. The acceleration terms are thus composed of a time-varying acceleration magnitude, $a(t)$, as in Equation 160. This magnitude increases with time, and its direction is defined by the in-plane and out-of-plane thrust angles, $\alpha$ and $\phi$, similar to the angles defined in Figure 4, but where $\alpha$ in this case is measured in the opposite direction from the $\hat{y}$ axis. These two angles are the control variables and are bounded as follows:

$$\alpha \in [-\pi, \pi] \tag{301}$$

$$\phi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]. \tag{302}$$

Thus, the acceleration terms can be described by,

$$a_x = a(f)\sin\alpha(f)\cos\phi(f) \tag{303}$$

$$a_y = a(f)\cos\alpha(f)\cos\phi(f) \tag{304}$$

$$a_z = a(f)\sin\phi(f), \tag{305}$$

where the acceleration magnitude (and the angles) are now functions of the true anomaly, $f$, of the reference orbit since that is the independent variable when using the Tschauner-Hempel equations of motion. When these acceleration terms are retained in the development of the Tschauner-Hempel equations of motion, they become multiplied by a scalar coefficient, like in [88],

$$B(f) = \frac{p^3 a(f)}{\mu k(f)^3}, \tag{306}$$

where $a(f)$ has been included in the scalar coefficient. With the defined acceleration terms and this coefficient, the Tschauner-Hempel equations of motion become,

$$\tilde{x}'' = 2\tilde{z}' + B(f)\sin\alpha\cos\phi \tag{307}$$

$$\tilde{y}'' = -\tilde{y} + B(f)\cos\alpha\cos\phi \tag{308}$$

$$\tilde{z}'' = \frac{3\tilde{z}}{k(f)} - 2\tilde{x}' + B(f)\sin\phi, \tag{309}$$

where it is understood that the control variables and Tschauner-Hempel states, $\tilde{X} = [\tilde{x}, \tilde{y}, \tilde{z}, \tilde{x}', \tilde{y}', \tilde{z}']^T$, are also functions of the true anomaly. These equations can be written in state space form as:

$$\tilde{X}' = A(f)\tilde{X} + B(f)U(\alpha, \phi), \tag{310}$$

where

$$U(\alpha, \phi) = \begin{bmatrix} 0 & 0 & 0 & \sin\alpha\cos\phi & \cos\alpha\cos\phi & \sin\phi \end{bmatrix}^T. \tag{311}$$

Both the pursuer and evader use Equation 310 as their equations of motion, where the dynamics of the pursuer will be represented with a subscript '$p$', and the evader

with a subscript '$e$':

$$\tilde{X}_p' = A(f)\tilde{X}_p + B(f)U_p \tag{312}$$

$$\tilde{X}_e' = A(f)\tilde{X}_e + B(f)U_e. \tag{313}$$

### 5.3 Differential Game Formulations using Indirect Heuristic Method

Given the equations of motion and control for both players, the differential games can now be formulated. Six main types of games are examined in this problem, where in all games the pursuer has a higher initial acceleration value than the evader, i.e. $a_{0_p} > a_{0_e}$, and both players have the same effective exhaust velocity. This ensures that the pursuer will always win, if given enough time.

For all games, the cost function is

$$J = f_f, \tag{314}$$

which is equivalent to the final time of the game. This means that the pursuer wishes to minimize the time to achieve a specific inspection goal, whereas the evader wishes to prolong that inspection goal as long as possible. This is a zero-sum game, since the sum of the two players' cost is zero.

Given the dynamic system of the two players, Equations 312–313, their initial conditions, the terminal constraints for each game,

$$\psi(\tilde{X}_p(f_f), \tilde{X}_e(f_f), f_f) = 0, \tag{315}$$

and the performance criterion, Equation 314, the goal is to find $u^*$ and $v^*$ such that

$$J(u^*, v) \leq J(u^*, v^*) \leq J(u, v^*), \tag{316}$$

where

$$u = \begin{bmatrix} \alpha_p & \phi_p \end{bmatrix}^T \tag{317}$$

$$v = \begin{bmatrix} \alpha_e & \phi_e \end{bmatrix}^T. \tag{318}$$

Following the indirect heuristic method (IHM), it is desired to find the game-theoretic saddle point by first applying the first-order necessary conditions. The Hamiltonian for all games considered is,

$$\mathcal{H} = \lambda_p^T \tilde{X}_p' + \lambda_e^T \tilde{X}_e' = \lambda_p^T \left[ A(f)\tilde{X}_p + B(f)U_p \right] + \lambda_e^T \left[ A(f)\tilde{X}_e + B(f)U_e \right], \tag{319}$$

where $\lambda_p$ and $\lambda_e$ are the vectors of costates,

$$\lambda_p = [\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6]^T \tag{320}$$

$$\lambda_e = [\lambda_7, \lambda_8, \lambda_9, \lambda_{10}, \lambda_{11}, \lambda_{12}]^T. \tag{321}$$

The costate equations for all games are

$$\lambda_p' = -A(f)^T \lambda_p \tag{322}$$

$$\lambda_e' = -A(f)^T \lambda_e, \tag{323}$$

where like Stupik [75, 74], it is desirable to find the state transition matrix for the costate equations. Since the YA STM, $\Theta$ in Equation 16, already exists for the system

$$\tilde{X}' = A(f)\tilde{X}, \tag{324}$$

then it can be used to find the STM for the costates equations when $A(f)$ is $-A(f)^T$.

The STM for the costate equations, $\Xi$, can be found symbolically by using MATLAB's Symbolic Toolbox via:

$$\Xi(f, f_0) = \left(\Theta(f, f_0)^{-1}\right)^T. \tag{325}$$

The optimal control for each player must satisfy,

$$\begin{bmatrix} u^{*^T} & v^{*^T} \end{bmatrix}^T = \arg\max_v \min_u \mathcal{H}, \tag{326}$$

since the controls are bounded. It can be seen that the Hamiltonian is separable, thus

$$u^* = \begin{bmatrix} \alpha_p^* & \phi_p^* \end{bmatrix}^T = \arg\min_{(\alpha_p, \phi_p)} \left(\lambda_4 \sin\alpha_p \cos\phi_p + \lambda_5 \cos\alpha_p \cos\phi_p + \lambda_6 \sin\phi_p\right). \tag{327}$$

This can be written as

$$u^* = \begin{bmatrix} \alpha_p^* & \phi_p^* \end{bmatrix}^T = \arg\min_{(\alpha_p, \phi_p)} \left(\cos\phi_p(\lambda_4 \sin\alpha_p + \lambda_5 \cos\alpha_p) + \lambda_6 \sin\phi_p\right), \tag{328}$$

which is in a form where Isaacs's Lemma on Circular Vectograms [64] may be applied to obtain:

$$\sin\alpha_p^* = \frac{-\lambda_4}{\sqrt{\lambda_4^2 + \lambda_5^2}} \tag{329}$$

$$\cos\alpha_p^* = \frac{-\lambda_5}{\sqrt{\lambda_4^2 + \lambda_5^2}} \tag{330}$$

$$\sin\phi_p^* = \frac{-\lambda_6}{\sqrt{\lambda_4^2 + \lambda_5^2 + \lambda_6^2}}. \tag{331}$$

Similarly, the optimal control equations for the evader can be obtained from

$$v^* = \begin{bmatrix} \alpha_e^* & \phi_e^* \end{bmatrix} = \arg\max_{(\alpha_e, \phi_e)} \left(\cos\phi_e(\lambda_{10} \sin\alpha_e + \lambda_{11} \cos\alpha_e) + \lambda_{12} \sin\phi_e\right), \tag{332}$$

and thus

$$\sin \alpha_e^* = \frac{\lambda_{10}}{\sqrt{\lambda_{10}^2 + \lambda_{11}^2}} \tag{333}$$

$$\cos \alpha_e^* = \frac{\lambda_{11}}{\sqrt{\lambda_{10}^2 + \lambda_{11}^2}} \tag{334}$$

$$\sin \phi_e^* = \frac{\lambda_{12}}{\sqrt{\lambda_{10}^2 + \lambda_{11}^2 + \lambda_{12}^2}}. \tag{335}$$

These optimal control equations are the same for all games examined in this problem.

The costate boundary conditions are different for each game considered. For each game, the function $\Phi$ is generated,

$$\Phi = f_f + \nu^T \psi, \tag{336}$$

where $\psi$ is different for each game, or inspection goal. The next section describes the specific terminal constraints and costate boundary conditions for each game. To use the IHM, according to Pontani [33], the costate boundary conditions must be homogeneous with respect to the costates in conjunction with the homogeneity of the costate equations. The developed costate equations, Equations 322–323, are homogeneous, and thus it is desirable that the costate boundary conditions developed in the next section are homogeneous as well. If these conditions apply, then Pontani claims that the transversality condition ($\mathcal{H}_f + 1 = 0$ for all games considered) is ignorable, and if an initial value of $\lambda$ for the pursuer and evader can be found such that $\lambda_0 = k_\lambda \lambda_0^*$, ($k_\lambda > 0$), then the same proportionality holds between $\lambda$ and the optimal $\lambda^*$ at any time. And, since the optimal control equations depend only on the relative magnitude of the costates and not the optimal costates themselves, then by satisfying all of the necessary conditions except for the transversality condition, the optimal control law can be found, with a scaled version of the optimal costates.

In addition to the homogeneity of the boundary conditions, it is also desirable that the terminal constraints (developed in the next section) are linear with respect to the states, as this simplifies the resulting boundary value problem and makes the problem more tractable to solve.

For most games examined, as will be seen, the following property holds, as in [74, 75],

$$\lambda_{p_f} + \lambda_{e_f} = 0. \tag{337}$$

Using the STM for the costates, it can be seen that,

$$\lambda_{p_f} + \lambda_{e_f} = 0 = \Xi\lambda_{p_0} + \Xi\lambda_{e_0} = \Xi(\lambda_{p_0} + \lambda_{e_0}), \tag{338}$$

which, assuming $\Xi$ is invertible, means that

$$\lambda_{p_0} = -\lambda_{e_0}, \tag{339}$$

and thus at any true anomaly,

$$\lambda_p = -\lambda_e, \tag{340}$$

meaning that the pursuer and evader costates are always equal and opposite. Plugging this relationship into the optimal control equations, it can be seen that, as shown in [74, 75],

$$\alpha_p^* = \alpha_e^* \tag{341}$$

$$\phi_p^* = \phi_e^*. \tag{342}$$

## 5.4 Game Conditions

For each game, a different set of terminal constraints is considered. Therefore, the following subsections present the terminal constraints for each game and the corresponding costate boundary conditions, which are all homogeneous and allow the use of the IHM.

### 5.4.1 Intercept.

In the intercept game, the goal of the pursuer is to match the final position of the evader in minimum time, while the goal of the evader is to delay that condition as long as possible. Thus, the vector of terminal constraints is simply:

$$\psi = \begin{bmatrix} \tilde{x}_p - \tilde{x}_e \\ \tilde{y}_p - \tilde{y}_e \\ \tilde{z}_p - \tilde{z}_e \end{bmatrix}, \tag{343}$$

and the costate boundary conditions become

$$\begin{aligned} \lambda_{1_f} &= \nu_1 & \lambda_{7_f} &= -\nu_1 \\ \lambda_{2_f} &= \nu_2 & \lambda_{8_f} &= -\nu_2 \\ \lambda_{3_f} &= \nu_3 & \lambda_{9_f} &= -\nu_3 \\ \lambda_{4_f} &= 0 & \lambda_{10_f} &= 0 \\ \lambda_{5_f} &= 0 & \lambda_{11_f} &= 0 \\ \lambda_{6_f} &= 0 & \lambda_{12_f} &= 0. \end{aligned} \tag{344}$$

If given $\lambda_{1_f} - \lambda_{3_f}$, all final costates can be determined.

### 5.4.2 Rendezvous.

In the rendezvous game, the goal of the pursuer is to match the final position *and* velocity of the evader in minimum time, while the goal of the evader is to delay that condition as long as possible. Thus, the vector of terminal constraints is simply:

$$\psi = \begin{bmatrix} \tilde{x}_p - \tilde{x}_e \\ \tilde{y}_p - \tilde{y}_e \\ \tilde{z}_p - \tilde{z}_e \\ \tilde{x}'_p - \tilde{x}'_e \\ \tilde{y}'_p - \tilde{y}'_e \\ \tilde{z}'_p - \tilde{z}'_e \end{bmatrix}, \tag{345}$$

and the costate boundary conditions become

$$\begin{aligned} \lambda_{1_f} &= \nu_1 \quad \lambda_{7_f} = -\nu_1 \\ \lambda_{2_f} &= \nu_2 \quad \lambda_{8_f} = -\nu_2 \\ \lambda_{3_f} &= \nu_3 \quad \lambda_{9_f} = -\nu_3 \\ \lambda_{4_f} &= \nu_4 \quad \lambda_{10_f} = -\nu_4 \\ \lambda_{5_f} &= \nu_5 \quad \lambda_{11_f} = -\nu_5 \\ \lambda_{6_f} &= \nu_6 \quad \lambda_{12_f} = -\nu_6. \end{aligned} \tag{346}$$

If given $\lambda_{1_f} - \lambda_{6_f}$, all final costates can be determined.

### 5.4.3 Obtain Sun Vector.

In this game, the goal of the pursuer is to align its position with the vector from the Sun to the evader at a point between the Sun and the evader such that the evader is lit with respect to the pursuer, in minimum time. The goal of the evader is to delay that condition as long as possible. Figure 51 shows the general terminal

conditions for this game, where the goal of the pursuer is to end on the vector from the Sun to the evader, $r_{se}$, at some distance from the evader, thus at $\kappa r_{se}$, where $\kappa \in (0, 1)$ and should be very close to one. This formulation is linear with respect

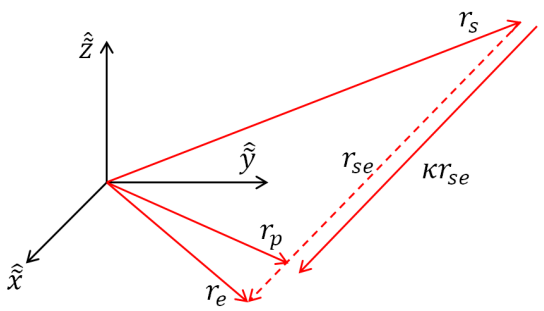



**Figure 51. General Terminal Conditions for Obtaining the Sun Vector**

to the states. However, the final range cannot be exactly prescribed, and some error must be allowed from the prescribed final range. For example, if a final range of $d$ km is desired between the pursuer and evader, then $\kappa$ may be estimated as

$$\kappa = \frac{-d||r_s(f_0)||_2 + ||r_s(f_0)||_2^2}{||r_s(f_0)||_2^2}, \tag{347}$$

where $r_s$ is the vector to the Sun in the relative frame. This should produce a final range relatively close to $d$ as long as the motion of the pursuer and evader remain close to the evader's original orbit (the reference orbit), and the game does not last for too long with respect to the period of the reference orbit. Given an estimate for $\kappa$ to achieve $d$, then the terminal constraint may be expressed as

$$r_p = r_s + \kappa(r_{se}), \tag{348}$$

where $r_p$ is the relative position of the pursuer, $r_e$ is the relative position of the evader, and $r_{se} = r_e - r_s$. Thus, in standard form, the vector of terminal constraints is

$$\psi = \begin{bmatrix} \tilde{x}_p - \tilde{x}_s - \kappa(\tilde{x}_e - \tilde{x}_s) \\ \tilde{y}_p - \tilde{y}_s - \kappa(\tilde{y}_e - \tilde{y}_s) \\ \tilde{z}_p - \tilde{z}_s - \kappa(\tilde{z}_e - \tilde{z}_s) \end{bmatrix}, \tag{349}$$

where $r_s$ does not depend on the pursuer and evader states and can be calculated for any given final true anomaly via Vallado's *sun* algorithm [13]. This formulation of the terminal constraints produces the following costate boundary conditions,

$$\begin{aligned} \lambda_{1_f} &= \nu_1 & \lambda_{7_f} &= -\kappa\nu_1 \\ \lambda_{2_f} &= \nu_2 & \lambda_{8_f} &= -\kappa\nu_2 \\ \lambda_{3_f} &= \nu_3 & \lambda_{9_f} &= -\kappa\nu_3 \\ \lambda_{4_f} &= 0 & \lambda_{10_f} &= 0 \\ \lambda_{5_f} &= 0 & \lambda_{11_f} &= 0 \\ \lambda_{6_f} &= 0 & \lambda_{12_f} &= 0, \end{aligned} \tag{350}$$

which are not a function of the players' states or the position of the Sun, thus simplifying the resulting boundary value problem. If given $\lambda_{1_f} - \lambda_{3_f}$, all final costates can be determined.

#### 5.4.3.1 Obtain Sun Vector at Exact Range.

This game condition is the same as the previous, but where $\kappa$ can be changed in an iterative fashion in an outer loop if the resulting final range is not close enough to the value desired. The game can be solved multiple times until some tolerance is reached and the resulting $\kappa$ value produces the desired terminal range between the pursuer and evader.

### 5.4.4 Match Energy.

The match energy game is where the goal of the pursuer is to match the energy of the evader in minimum time, such that if each player stopped thrusting at the end of the game, then the relative motion between the two players would remain bounded after one period. The goal of the evader is to delay that condition as long as possible. To ensure the energy of the two players is matched, the change in the semi-major axes of both players should be the same. Thus, using Equation 32 for each player, the following should apply,

$$\delta a_p = \frac{2a}{\eta^2} c_{3_p} = \delta a_e = \frac{2a}{\eta^2} c_{3_e}, \tag{351}$$

where $a$ and $\eta$ are those of the reference orbit, and the coefficients $c_{3_p}$ and $c_{3_e}$ from Equation 40 are referenced to the reference orbit as well, but are functions of the relative states of the pursuer and evader respectively. Expressing the terms multiplied by the pursuer or evader states inside Equations 38–41 as $b_{ij}$, where $i = 1, ..., 4$ for each $c_i$ and where $j = 1, ...3$ for the three terms multiplied by the relative states inside each $c_i$, the vector of terminal constraint can be written as,

$$\psi = \left[ b_{31}(\tilde{x}_p - \tilde{x}_e) + b_{32}(\tilde{x}'_p - \tilde{x}'_e) + b_{33}(\tilde{y}'_p - \tilde{y}'_e) \right], \tag{352}$$

where each $b_{ij}$ is a function of the final true anomaly. This formulation of the terminal constraints produces the following costate boundary conditions,

$$\begin{aligned}
\lambda_{1_f} &= b_{31}\nu_1 \quad & \lambda_{7_f} &= -b_{31}\nu_1 \\
\lambda_{2_f} &= 0 & \lambda_{8_f} &= 0 \\
\lambda_{3_f} &= 0 & \lambda_{9_f} &= 0 \\
\lambda_{4_f} &= b_{32}\nu_1 & \lambda_{10_f} &= -b_{32}\nu_1 \\
\lambda_{5_f} &= b_{33}\nu_1 & \lambda_{11_f} &= -b_{33}\nu_1 \\
\lambda_{6_f} &= 0 & \lambda_{12_f} &= 0.
\end{aligned} \tag{353}$$

If given $\lambda_{1_f}$ and the final true anomaly, all final costates can be determined.

### 5.4.5 Obtain Sun Vector and Match Energy.

In this game, the goal of the pursuer is a combination of two previous conditions — to align itself with the Sun vector at an approximate range while also matching the energy of the evader in minimum time — while the goal of the evader is to delay that combination of conditions as long as possible. The vector of terminal constraints is thus,

$$\psi = \begin{bmatrix} \tilde{x}_p - \tilde{x}_s - \kappa(\tilde{x}_e - \tilde{x}_s) \\ \tilde{y}_p - \tilde{y}_s - \kappa(\tilde{y}_e - \tilde{y}_s) \\ \tilde{z}_p - \tilde{z}_s - \kappa(\tilde{z}_e - \tilde{z}_s) \\ b_{31}(\tilde{x}_p - \tilde{x}_e) + b_{32}(\tilde{x}'_p - \tilde{x}'_e) + b_{33}(\tilde{y}'_p - \tilde{y}'_e) \end{bmatrix}, \tag{354}$$

and the costate boundary conditions become,

$$\begin{aligned}
\lambda_{1_f} &= \nu_1 + b_{31}\nu_4 & \lambda_{7_f} &= -\kappa\nu_1 - b_{31}\nu_4 \\
\lambda_{2_f} &= \nu_2 & \lambda_{8_f} &= -\kappa\nu_2 \\
\lambda_{3_f} &= \nu_3 & \lambda_{9_f} &= -\kappa\nu_3 \\
\lambda_{4_f} &= b_{32}\nu_4 & \lambda_{10_f} &= -b_{32}\nu_4 \\
\lambda_{5_f} &= b_{33}\nu_4 & \lambda_{11_f} &= -b_{33}\nu_4 \\
\lambda_{6_f} &= 0 & \lambda_{12_f} &= 0.
\end{aligned} \tag{355}$$

If given $\lambda_{1_f} - \lambda_{4_f}$ and the final true anomaly, all final costates can be determined.

### 5.4.6 Match Energy and Remain Close.

In this last game considered, the goal of the pursuer is to both match the energy of the evader and also remain within a prescribed range of the evader during the ensuing period, and to accomplish these goals in minimum time, while the goal of the evader is to delay that combination of conditions as long as possible. In order to remain close during the ensuing period, Sengupta's relative orbit parameters for periodic, elliptical relative motion, $\varrho_1$, $\varrho_2$, and $\varrho_3$ in Equations 24–26, can be constrained to place the pursuer into the same orbit as the evader, at a prescribed distance away, and with an amplitude for any desired relative cross-track motion.

To do this, the same energy matching constraint must be enforced as before, namely Equation 352. Next, $\varrho_1$ must be set equal to zero, where this is one of the two relative motion size parameters for the motion of the pursuer relative to the evader, *not* relative to the reference orbit. Thus, examining Equation 27, $\delta e$ and $\delta M_0$ of the pursuer relative to the evader must both be equal to zero. Therefore, $\delta e_p$ and $\delta e_e$ (the change in eccentricity of each player with respect to the reference orbit or the

original orbit of the evader) must be equal. Therefore, the second constraint is

$$\delta e_p - \delta e_e = -\eta^2(c_{1_p} - c_{1_e}) = 0. \tag{356}$$

Likewise, $\delta M_{0_p}$ and $\delta M_{0_e}$ must be equal, and the third constraint is

$$\delta M_{0_p} - \delta M_{0_e} = \frac{\eta^3}{e}(c_{2_p} - c_{2_e}) = 0. \tag{357}$$

These two constraints ensure that $\varrho_1 = 0$ for the motion of the pursuer with respect to the evader. Next, $\frac{\varrho_2}{p_e}$ (where $p_e$ is the semi-latus rectum of the evader's final orbit since this is for the motion of the pursuer relative to the evader) must be set equal to the desired in-track placement of the pursuer relative to the evader, or $d_y$. Therefore, the following condition must apply,

$$\varrho_2 = p_e d_y. \tag{358}$$

It can be seen by examining Equation 28, that since $\delta M_0$ of the pursuer relative to the evader is already equal to zero, then a way to enforce Equation 358 is to set $\delta\Omega$ of the pursuer relative to the evader equal to zero, and then the following must apply,

$$\varrho_2 = p_e \left(\delta\omega_{\text{o}_p} - \delta\omega_{\text{o}_e}\right). \tag{359}$$

Therefore, setting this equation equal to Equation 358,

$$\delta\omega_{\text{o}_p} - \delta\omega_{\text{o}_e} = d_y, \tag{360}$$

and thus the fourth constraint, using Equation 37, is

$$\begin{aligned}\delta\omega_{\text{œ}_p} - \delta\omega_{\text{œ}_e} &= \left(c_{4_p} - \frac{\delta M_{0_p}}{\eta^3} - \delta\Omega_p \cos i\right) - \left(c_{4_e} - \frac{\delta M_{0_e}}{\eta^3} - \delta\Omega_e \cos i\right) \\ &= c_{4_p} - c_{4_e} = d_y. \end{aligned} \tag{361}$$

As mentioned, $\delta\Omega$ of the pursuer relative to the evader must also set equal to zero, thus the fifth constraint is

$$\delta\Omega_p - \delta\Omega_e = 0, \tag{362}$$

or,

$$\cos(\omega_{\text{œ}} + f)(\tilde{z}_e - \tilde{z}_p) + \sin(\omega_{\text{œ}} + f)(\tilde{z}'_p - \tilde{z}'_e) = 0. \tag{363}$$

And finally, the sixth and final constraint prescribes the amplitude of the ensuing cross-track motion of the pursuer relative to the evader. To do this, $\frac{\varrho_3}{p_e}$ must be set equal to the desired amplitude, or $d_z$. Therefore,

$$\varrho_3 = p_e d_z, \tag{364}$$

and by examining Equation 29 and realizing that $\delta\Omega$ of the pursuer relative to the evader is already set equal to zero, then the following condition must apply,

$$\varrho_3 = p_e(\delta i_p - \delta i_e). \tag{365}$$

Thus, setting the two previous equations equal to one another,

$$\delta i_p - \delta i_e = d_z, \tag{366}$$

and so the sixth and final constraint can be written as

$$\sin(\omega_{\text{œ}} + f)(\tilde{z}_p - \tilde{z}_e) + \cos(\omega_{\text{œ}} + f)(\tilde{z}'_p - \tilde{z}'_e) - d_z = 0. \tag{367}$$

The vector of terminal equality constraints can thus be written as

$$\psi = \begin{bmatrix} b_{31}(\tilde{x}_p - \tilde{x}_e) + b_{32}(\tilde{x}'_p - \tilde{x}'_e) + b_{33}(\tilde{y}'_p - \tilde{y}'_e) \\ b_{11}(\tilde{x}_p - \tilde{x}_e) + b_{12}(\tilde{x}'_p - \tilde{x}'_e) + b_{13}(\tilde{y}'_p - \tilde{y}'_e) \\ b_{21}(\tilde{x}_p - \tilde{x}_e) + b_{22}(\tilde{x}'_p - \tilde{x}'_e) + b_{23}(\tilde{y}'_p - \tilde{y}'_e) \\ b_{41}(\tilde{x}_p - \tilde{x}_e) + b_{42}(\tilde{x}'_p - \tilde{x}'_e) + b_{43}(\tilde{y}'_p - \tilde{y}'_e) + \tilde{y}_p - \tilde{y}_e - d_y \\ \cos(\omega_{\text{œ}} + f)(\tilde{z}_e - \tilde{z}_p) + \sin(\omega_{\text{œ}} + f)(\tilde{z}'_p - \tilde{z}'_e) \\ \sin(\omega_{\text{œ}} + f)(\tilde{z}_p - \tilde{z}_e) + \cos(\omega_{\text{œ}} + f)(\tilde{z}'_p - \tilde{z}'_e) - d_z \end{bmatrix}. \tag{368}$$

These constraints ensure that the pursuer matches the orbit of the evader at a prescribed in-track distance away, $d_y$, and with a relative cross-track amplitude of $d_z$. They have been developed to be linear with respect to the states such that the resulting boundary value problem is more tractable to solve. The corresponding costate boundary conditions are

$$\begin{aligned} \lambda_{1_f} &= b_{31}\nu_1 + b_{11}\nu_2 + b_{21}\nu_3 + b_{41}\nu_4 & \lambda_{7_f} &= -\lambda_{1_f} \\ \lambda_{2_f} &= \nu_4 & \lambda_{8_f} &= -\lambda_{2_f} \\ \lambda_{3_f} &= -\cos(\omega_{\text{œ}} + f)\nu_5 + \sin(\omega_{\text{œ}} + f)\nu_6 & \lambda_{9_f} &= -\lambda_{3_f} \\ \lambda_{4_f} &= b_{32}\nu_1 + b_{12}\nu_2 + b_{22}\nu_3 + b_{42}\nu_4 & \lambda_{10_f} &= -\lambda_{4_f} \\ \lambda_{5_f} &= b_{33}\nu_1 + b_{13}\nu_2 + b_{23}\nu_3 + b_{43}\nu_4 & \lambda_{11_f} &= -\lambda_{5_f} \\ \lambda_{6_f} &= \sin(\omega_{\text{œ}} + f)\nu_5 + \cos(\omega_{\text{œ}} + f)\nu_6 & \lambda_{12_f} &= -\lambda_{6_f}. \end{aligned} \tag{369}$$

If given $\lambda_{1_f} - \lambda_{6_f}$ and the final true anomaly, all final costates can be determined.

## 5.5 Boundary Value Problem Formulations via Indirect Heuristic Method

Following the IHM, each boundary value problem resulting from each game condition is solved by formulating a parameter optimization problem to be solved by a metaheuristic method, namely MATLAB's PSO or GA, where the purpose of the optimizer is solely to satisfy the terminal constraints contained in $\psi$ for each game. For all games, the minimum number of final costates required to determine all final costates, (which is equal to the number of terminal constraints, $m$), become optimization variables. Stupik, et al. [74, 75] used initial costates; however, the final costates are used here to make the problem simpler to formulate and solve. Since the conditions necessary to apply the IHM are satisfied, only the relative magnitude of the costates are of importance, and thus following the IHM, the $m$ final costates for each problem can be bounded as follows,

$$\lambda_{i_f} \in [-1, 1] \qquad i = 1, ..., m. \tag{370}$$

The final time (which can be converted to the final true anomaly) is also included as an optimization variable in each problem, and is bounded,

$$t_f \in [0, t_{f_{max}}], \tag{371}$$

where $t_{f_{max}}$ should be large enough to ensure a solution can be found. The optimization variables for each problem must then be found to satisfy the terminal constraints. This is accomplished as follows:

1. Given the value of each of the minimum number of final costates required for the current problem, determine the rest of the final costates via the costate boundary condition equations for the current problem.

2. Propagate the costates backwards analytically from the given final time (final true anomaly) using the costate STM, $\Xi$, in Equation 325, to find the initial costates.

3. Propagate the states forward numerically (for example with MATLAB's *ode45*), using Equations 312-313, where at each time step the control is calculated via Equations 329–331 (and Equations 333–335 if necessary), which are functions of the costates and can be propagated forward analytically from the initial costates, again with $\Xi$.

4. Once the states have been propagated to the given final time (or final true anomaly), evaluate the terminal constraints contained in $\psi$ for the current problem.

5. Iterate (using the metaheuristic optimization algorithm), repeating steps 1.-4. until the prescribed tolerance is met.

6. Extract the differential game solution.

For each problem, the static optimization problem is solved with either MATLAB's PSO or MATLAB's GA, depending on the amount and type of constraints. If the PSO is used, the constraints are satisfied by summing the square of each constraint in the cost function

$$J_{PSO} = \sum_{i}^{m} \psi_i^2. \tag{372}$$

If the GA is used, then the cost function is set equal to zero,

$$J_{GA} = 0, \tag{373}$$

and the equality constraints are satisfied by using the non-default nonlinear constraint algorithm, *Penalty Algorithm*. Table 22 shows the following for each game: the number of equality constraints, $m$, contained in $\psi$, the optimization parameters, and the solver to be used.

**Table 22. Problem C (Game Optimization Problem) Formulations**

| Inspection Goal | Size of $\psi$ $(m)$ | Optimization Parameters | Solver |
|---|---|---|---|
| Intercept | 3 | $\chi = [\lambda_{1_f}, \lambda_{2_f}, \lambda_{3_f}, t_f]$ | PSO |
| Rendezvous | 6 | $\chi = [\lambda_{1_f}, \lambda_{2_f}, \lambda_{3_f}, \lambda_{4_f}, \lambda_{5_f}, \lambda_{6_f}, t_f]$ | GA |
| Obtain Sun Vector | 3 | $\chi = [\lambda_{1_f}, \lambda_{2_f}, \lambda_{3_f}, t_f]$ | PSO |
| Match Energy | 1 | $\chi = [\lambda_{1_f}, t_f]$ | PSO |
| Obtain Sun Vector & Match Energy | 4 | $\chi = [\lambda_{1_f}, \lambda_{2_f}, \lambda_{3_f}, \lambda_{4_f}, t_f]$ | GA |
| Match Energy & Remain Close | 6 | $\chi = [\lambda_{1_f}, \lambda_{2_f}, \lambda_{3_f}, \lambda_{4_f}, \lambda_{5_f}, \lambda_{6_f}, t_f]$ | GA |

## 5.6 Simulations and Results

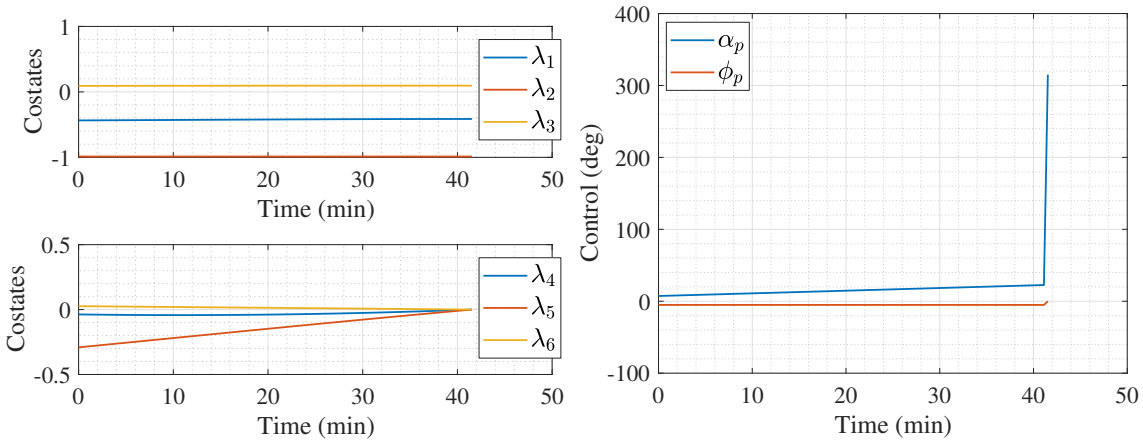
The solutions presented in this section use the simulation parameters shown in Table 23, where these include the initial conditions from [74, 75] in order to compare the intercept case when the eccentricity is set equal to zero. Note that $a_{0_p} > a_{0_e}$ to ensure the games can be completed.

**Table 23. Problem C Simulation Parameters**

| Time and Sun | Reference Orbit | Pursuer Properties | Evader Properties |
|---|---|---|---|
| $\text{Date}_0$=17 Mar 2018<br>$\text{Time}_0$=13:00.00<br>$\kappa = 0.9999999328126$ | $a_c = 42,164.137$ km<br>$e = 0.2$<br>$f_0 = 0^o$<br>$i = 5^o$<br>$\omega_{\text{œ}} = 10^o$<br>$\Omega = 45^o$ | $r_{p_0} = [-38.93, -100, 0]^T$ km<br>$v_{p_0} = [0, 0, 0]^T$ m/s<br>$a_{0_p} = 0.0686$ m/s$^2$<br>$c_p = 3$ km/s | $r_{e_0} = [0, 0, 0]^T$ km<br>$v_{e_0} = [0, 0, 0]^T$ m/s<br>$a_{0_e} = 0.0343$ m/s$^2$<br>$c_e = 3$ km/s |

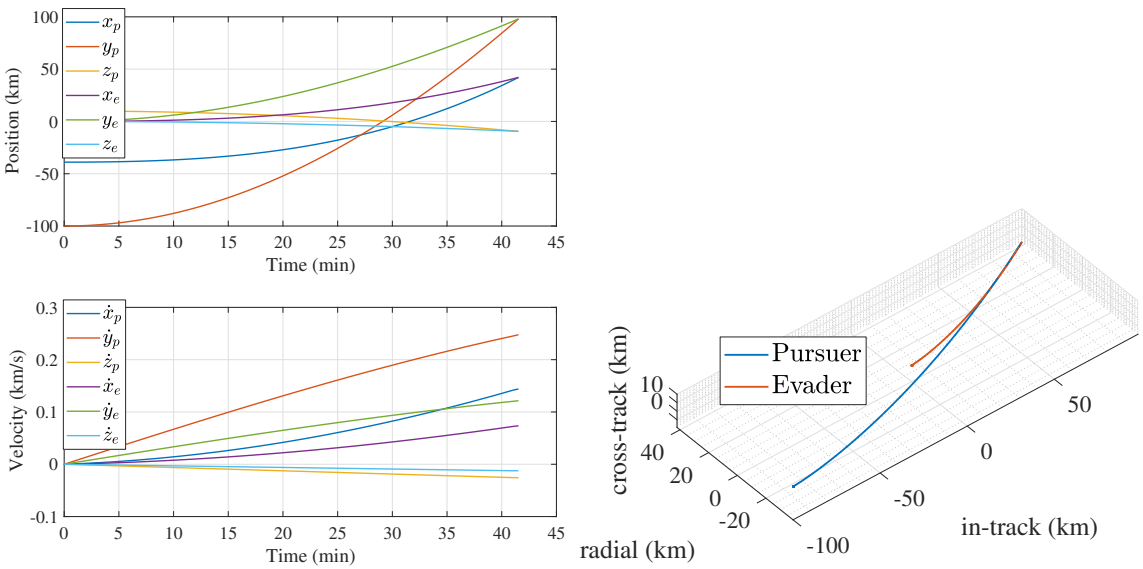
It was desired to compare the intercept solution with $e = 0$ to the first solution obtained in [74, 75] to see if they were approximately the same and to verify that the solution collapsed to the HCW solution when the eccentricity was set equal to zero. Indeed, the same solution was obtained, with a final time of $t_f = 41.32$ minutes and the same final position. This solution took 38.5 seconds to compute, where

all solutions obtained in this section use MATLAB's parallel processing with four processors. The value of the cost function, or the sum of the constraint violations squared was $1.15 \times 10^{-8}$. Since the solutions match, they are not shown here.




(a) Pursuer Costates (Negative for Evader) (b) Pursuer Control (Equal for Evader)

**Figure 52. Intercept Game: Costates and Control**

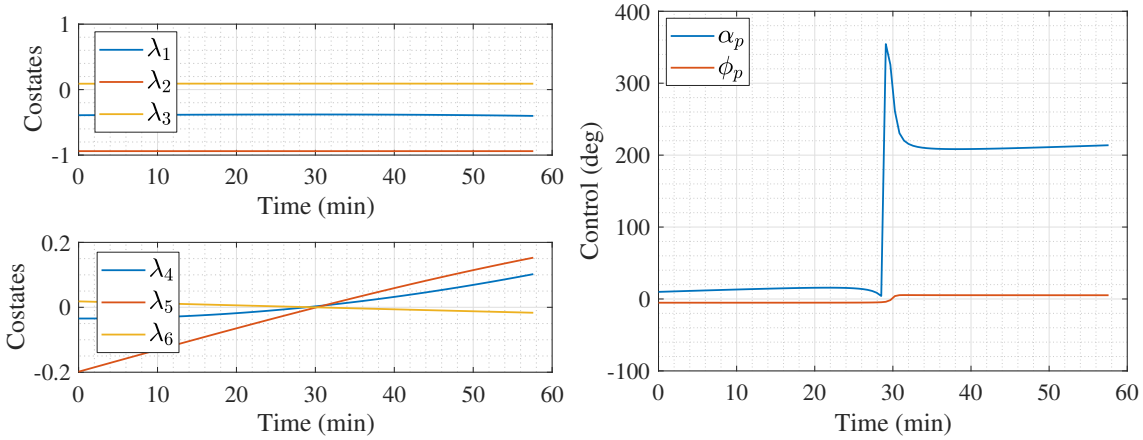



(a) States (b) 3-D Trajectory

**Figure 53. Intercept Game: States and Trajectory**

With the zero-eccentricity intercept solution matching [74, 75], the intercept game was run with $e = 0.2$, which the rest of the games in this section use to showcase

the ability to solve games that take place with respect to an elliptical reference orbit. The resulting costates and control are shown in Figure 52, where the evader costates are equal and opposite the pursuer costates, and the control for both players is the same. Note that the last control values in the plot are a result from the velocity costates being zero, and are invalid and not of importance. The states and trajectory are shown in Figure 53. This solution took 36.4 seconds to compute, with a final time of $t_f = 41.53$ minutes, and a PSO constraint violation of $3.79 \times 10^{-8}$.

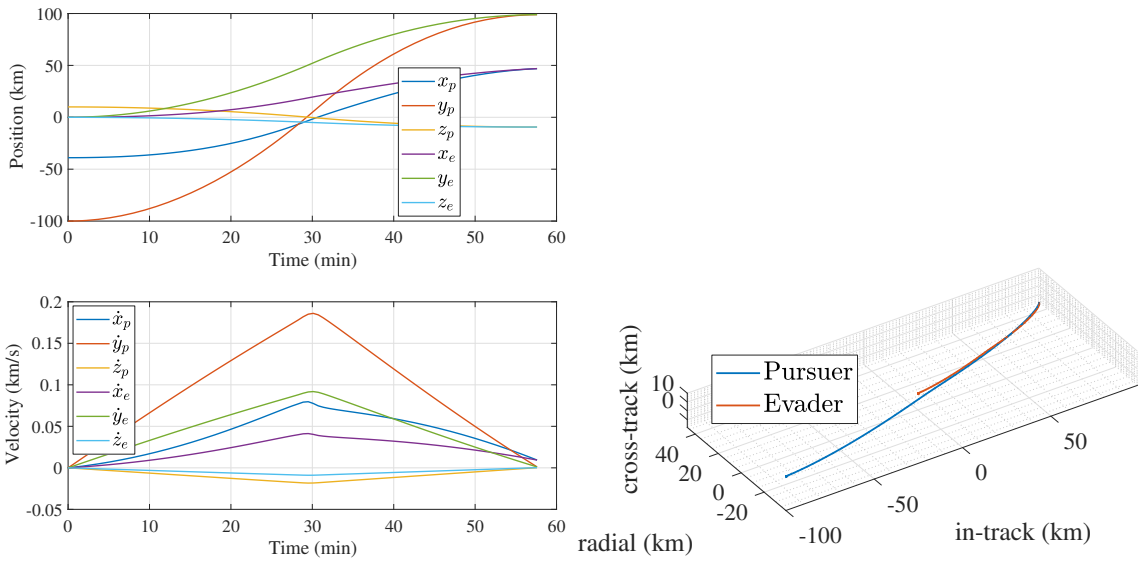



(a) Pursuer Costates (Negative for Evader) (b) Pursuer Control (Equal for Evader)

**Figure 54. Rendezvous Game: Costates and Control**

The rendezvous game costates and control are shown in Figure 54, where the evader costates are equal and opposite the pursuer costates, and the control for both players is the same. The states and trajectory are shown in Figure 55. The states take an interesting shape, where the optimal states for the evader prolong the rendezvous condition as long as possible. This solution took 8.64 minutes to compute since the GA was used instead of the PSO. The final time is $t_f = 57.61$ minutes, and the maximum GA constraint violation is $5.12 \times 10^{-10}$.

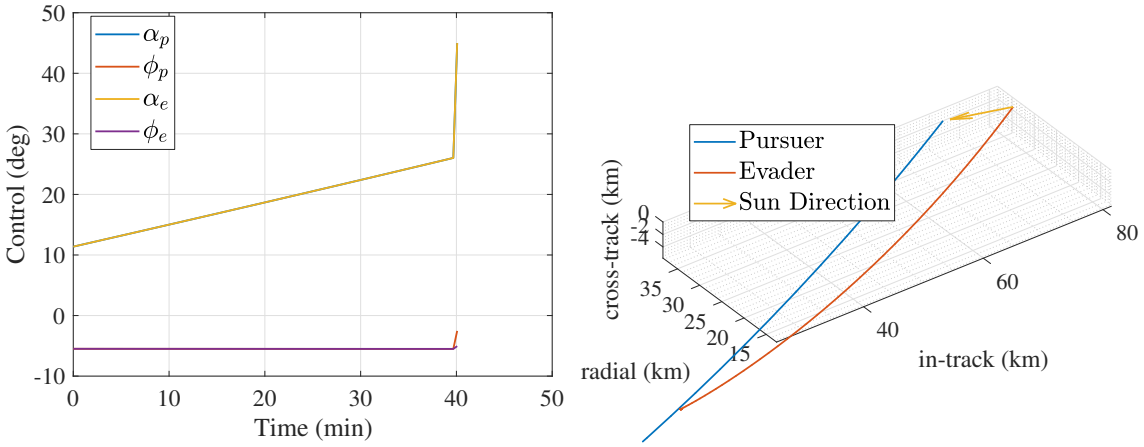
The obtain Sun vector game control and trajectory are shown in Figure 56. The control for both players are approximately the same, but not exactly, since $\lambda_{p_f} + \lambda_{e_f} \neq 0$. For this game, the desired range at the end of the game was set to $d = 10$ km. Using




(a) States (b) 3-D Trajectory

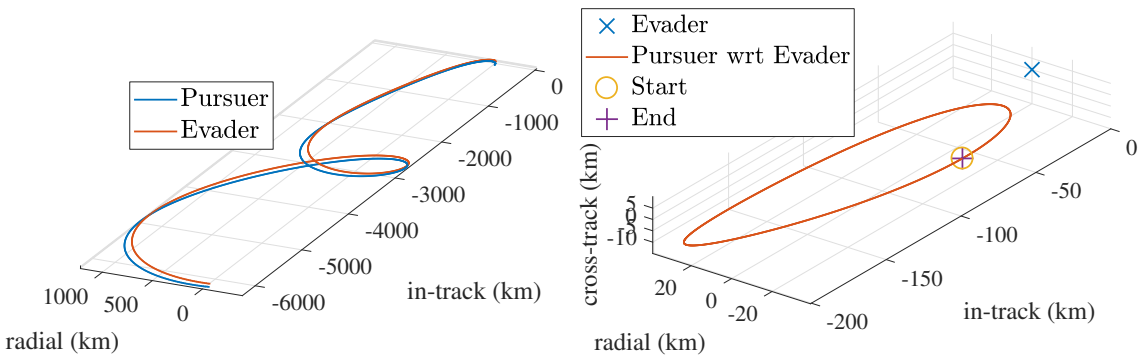
**Figure 55. Rendezvous Game: States and Trajectory**

Equation 347 for $\kappa$ to obtain $d$, the resulting range is 10.0006 km. The trajectory shows how at the end of the game, the pursuer is approximately 10 km away from the evader, at a point along the Sun vector with respect to the evader, such that the evader is illuminated with respect to the pursuer. This solution took 51.3 seconds to compute, with a final time $t_f = 40.08$ minutes, and a PSO constraint violation of $6.97 \times 10^{-7}$.




(a) Pursuer and Evader Control (b) 3-D Trajectory

**Figure 56. Obtain Sun Vector Game: Control and Trajectory**




(a) 3-D Trajectory

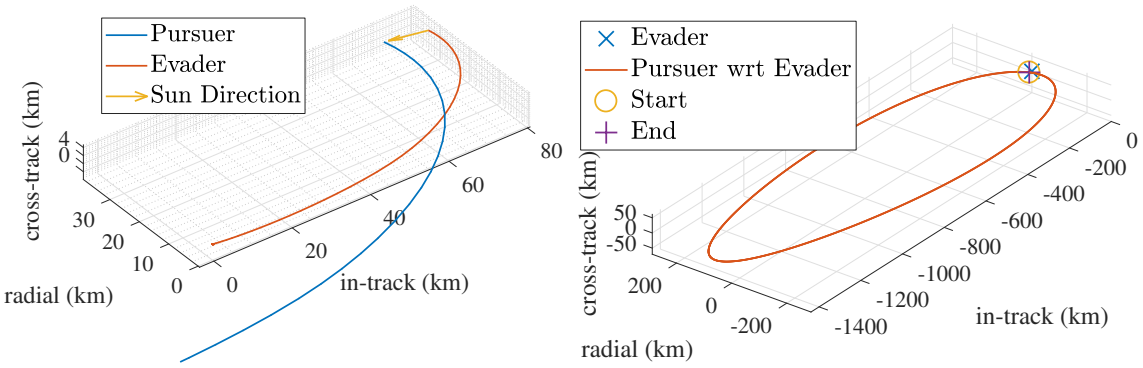
(b) Pursuer Relative to Evader

**Figure 57. Match Energy Game: Trajectories After Game Concludes**

The match energy game ends relatively quickly, with a final time of $t_f = 3.86$ minutes. This is quite possible, since no other constraint is being enforced and the pursuer simply needs to match the semi-major axis of the evader. The resulting motion right after the game concludes is shown in Figure 57, where (a) shows the relative positions of both players with respect to the original reference orbit, and (b) shows the position of the pursuer with respect to the evader over the course of a couple orbits. Thus, as can be seen, the two players travel far from the original reference orbit. Also, they don't necessarily remain close to each other, but the motion of the pursuer is bounded with respect to the evader since the energy has been matched. This solution took just 13.2 seconds to compute, with a PSO constraint violation of $1.00 \times 10^{-10}$.

The obtain Sun vector *and* match energy game results are shown in Figure 58, where (a) shows that the desired lighting is obtained at the end of the game, and (b) shows that the energy has been matched. With the estimate for $\kappa$, the final range, 10.0008 km, is very close to the desired (10 km). It is interesting to note however that although the energy has been matched, the pursuer does not remain close to the evader throughout the ensuing motion, and gets even farther away from the evader than in the previous game. This solution took 2.73 minutes to compute, with a final

time of $t_f = 51.80$ minutes and a maximum GA constraint violation of $1.58 \times 10^{-10}$.

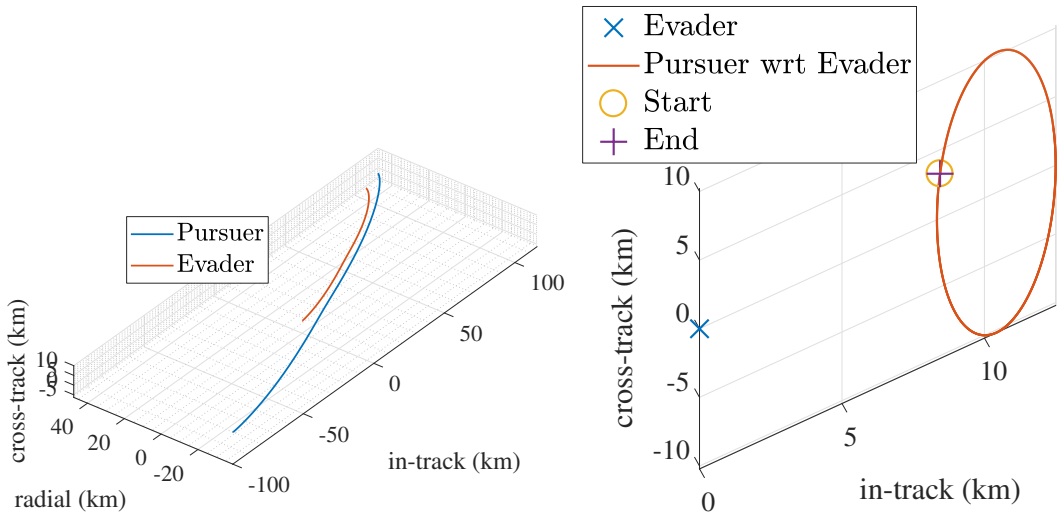



(a) 3-D Trajectory During Game (b) Pursuer Relative to Evader After Game

**Figure 58. Obtain Sun Vector & Match Energy Game: 3-D Trajectories**

For the match energy and remain close game, the desired in-track distance of the pursuer from the evader for the motion after the game concludes is $d_y = 10$ km, and the desired cross-track amplitude for the motion after the game ends is $d_z = 10$ km. The results can be seen in Figure 59, where (a) shows the trajectory during the game, and (b) shows that the desired motion after the game concludes has been achieved. Due to the elliptic nature of the orbit, the in-track separation distance oscillates throughout the orbit. This solution took 13.10 minutes to compute, with a final time of $t_f = 59.56$ minutes and a maximum GA constraint violation of $1.26 \times 10^{-10}$.

A summary of the simulation results is shown in Table 24. When using the PSO, the settings were left at their defaults, with the exception of using MATLAB's parallel processing as previously mentioned. When using the GA, in addition to using the parallel processing, the maximum number of generations was increased to 1,500, and the number of starting points was set to 200 which happened to be a non-default option for just one of the cases.




(a) 3-D Trajectory During Game (b) Pursuer w.r.t. Evader After Game

**Figure 59. Match Energy & Remain Close Game: 3-D Trajectories**

**Table 24. Problem C Simulation Results (using Table 23 parameters)**

| Game Condition | $t_f$ (min) | Solver | CPU Time | Constraint Violation |
|---|---|---|---|---|
| Intercept (with $e = 0$) | 41.32 | PSO | 38.5 sec | $1.15 \times 10^{-8}$ |
| Intercept | 41.53 | PSO | 36.4 sec | $3.79 \times 10^{-8}$ |
| Rendezvous | 57.61 | GA | 8.64 min | $5.12 \times 10^{-10}$ |
| Obtain Sun Vector | 40.08 | PSO | 51.3 sec | $6.97 \times 10^{-7}$ |
| Match Energy | 3.86 | PSO | 13.2 sec | $1.00 \times 10^{-10}$ |
| Obtain Sun Vector and Match Energy | 51.80 | GA | 2.73 min | $1.58 \times 10^{-10}$ |
| Match Energy and Remain Close | 59.56 | GA | 13.10 min | $1.26 \times 10^{-10}$ |

## 5.7 Solution Validation

This section is designed to provide some confidence that the solutions obtained in the previous section are in fact the differential game solutions. If a differential game solution has been found, that is, if $u^*$ and $v^*$ have been found such that

$$J(u^*, v) \leq J(u^*, v^*) \leq J(u, v^*), \tag{374}$$

then, if any suboptimal $v$ is used while $u^*$ is used, then the pursuer should be able to further minimize the performance index. Likewise, if any suboptimal $u$ is used while $v^*$ is used, then the evader should be able to further maximize the performance

index. Thus, this section uses that logic to formulate and solve one-sided optimal control problems, again using the IHM, where the control of one of the players is deterministic, or known beforehand, to add confidence that the differential game solutions have indeed been found. Both the intercept game and rendezvous game solutions are analyzed, where these two have been chosen to be analyzed since one used the PSO to solve the resulting boundary value problem and the other used the GA.

To analyze both the intercept game and rendezvous game solutions, multiple one-sided problems are formulated and solved. The first one-sided optimal control problem uses $v^*$ for the control of the evader and a problem is formulated to see if the pursuer can intercept (or rendezvous with) the evader any sooner than the final time obtained from the differential game solution. If the solution obtained in the previous section is indeed the differential game solution, then the final time from this one-sided optimal control problem should match. To do this for the intercept case, approximating polynomials are fit to $\tilde{x}_e^*$, $\tilde{y}_e^*$, and $\tilde{z}_e^*$ as functions of the true anomaly. This was done to avoid numerically propagating the deterministic position of the evader, and instead use the polynomial functions in the terminal constraint vector. For the rendezvous case, the approximating polynomials were not accurate enough, thus the evader state is numerically propagated based on the optimal evader costates found from the differential game. The objective for this first one-sided optimal control problem is to minimize the time to intercept (or rendezvous with) the deterministic position (or state) of the evader. Thus, the cost function is,

$$J = t_f, \tag{375}$$

subject to,

$$\psi = \begin{bmatrix} \tilde{x}_{p_f} - \tilde{x}_e^*(f_f) \\ \tilde{y}_{p_f} - \tilde{y}_e^*(f_f) \\ \tilde{z}_{p_f} - \tilde{z}_e^*(f_f) \end{bmatrix}, \tag{376}$$

for the intercept case and,

$$\psi = \begin{bmatrix} \tilde{x}_{p_f} - \tilde{x}_e^*(f_f) \\ \tilde{y}_{p_f} - \tilde{y}_e^*(f_f) \\ \tilde{z}_{p_f} - \tilde{z}_e^*(f_f) \\ \tilde{x}'_{p_f} - \tilde{x}_e'^*(f_f) \\ \tilde{y}'_{p_f} - \tilde{y}_e'^*(f_f) \\ \tilde{z}'_{p_f} - \tilde{z}_e'^*(f_f) \end{bmatrix}, \tag{377}$$

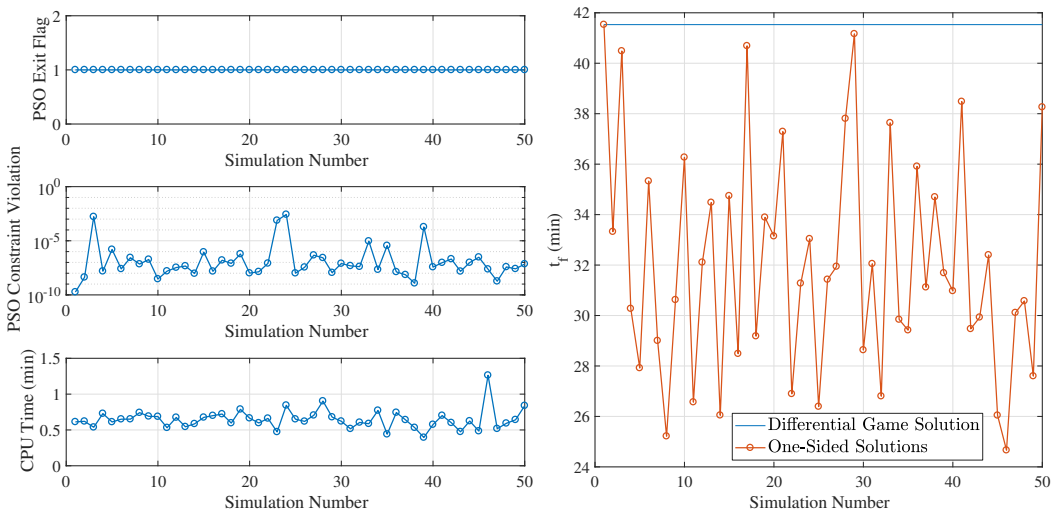
for the rendezvous case, where it is emphasized that the terminal states of the evader are solely functions of the given final true anomaly. These boundary conditions still allow for the use of the IHM for these one-sided optimal control problems because after applying the necessary conditions, the costate boundary conditions are the same as they were for the pursuer in the differential game for both the intercept and rendezvous cases. Applying the necessary conditions also produces the same control, costate, and state equations that the pursuer had in the differential games.

For the rest of the one-sided optimal control problems, the evader uses a control other than $v^*$. Thus, the pursuer, in a one-sided optimal control problem, should be able to intercept the evader sooner. If there is a $v$ which outperforms $v^*$, i.e. makes the pursuer take longer to intercept (or rendezvous with) the evader compared to the differential game solution, then the differential game solution was not found. For these one-sided optimal control problems where a $v$ other than $v^*$ is used for the evader, the

first strategy for $v$ is to point the thruster in the direction of $r_{p_0}$ such that the evader accelerates continually away from $r_{p_0}$. For the rest of the suboptimal $v$ strategies, the thruster is pointed in a constant, random direction during each of the problems. These formulations still allow the use of the IHM, and are developed to show that the pursuer can intercept the evader equal to or sooner than the differential game final time for any of the suboptimal strategies tested. For these one-sided optimal control problems, the evader states are numerically propagated from the initial conditions with the constant control laws. The cost function is the same as before, and the terminal constraints are the same except that the evader states at a given final true anomaly are not the optimal ones from the differential game solution. Again, for all these one-sided optimal control problems, the IHM can be used, and the same equations from applying the necessary equations are obtained as before.

The intercept game analysis results, where MATLAB's PSO is used to solve the resulting boundary value problems for the one-sided optimal control problems, can be seen in Figure 60. Figure 60 (a) shows the PSO exit flags, the constraint violations, and the CPU times for all fifty one-sided problems tested. As can be seen, constraints were satisfied for every problem, and most problems were solved in less than one minute. Figure 60 (b) shows the final time for each problem. The first final time obtained matches the differential game final time, as it should, since $v^*$ was used for the evader's control. The rest of the final times obtained are less than the differential game final time, as they should be, since a $v$ other than $v^*$ is used and the pursuer is able to further minimize the performance index.

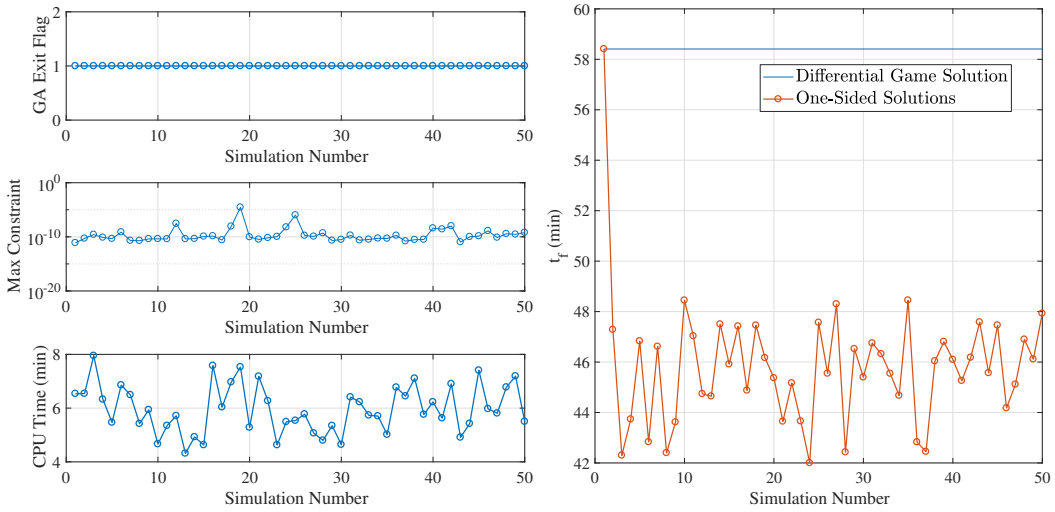
The rendezvous game analysis results, where MATLAB's GA is used to solve the resulting boundary value problems for the one-sided optimal control problems, can be seen in Figure 61. Figure 61 (a) shows the GA exit flags, the constraint violations, and the CPU times for all fifty one-sided problems. As can be seen, constraints were




(a) Exit Flag, Violation, & CPU Time (b) $t_f$ for Each One-Sided Problem

**Figure 60. Intercept Game: Solution Validation**

satisfied for every problem, and most problems were solved in about six minutes. Figure 61 (b) shows the final time for each problem. The first final time obtained matches the differential game final time, as it should. The rest of the final times obtained are less than the differential game final time, also as they should be.




(a) Exit Flag, Violation, & CPU Time (b) $t_f$ for Each One-Sided Problem

**Figure 61. Rendezvous Game: Solution Validation**

## 5.8 Problem C Conclusion

It is desirable to formulate differential games (and possibly some one-sided problems) such that the IHM can be used due to: a) no need for an initial guess for the costates, b) the ability to search for the final (or initial) costates in an arbitrary range, c) the relatively fast computation times, d) the global nature of the method, and e) the fact that an entire dynamic, differential game can be reduced to a static optimization problem composed of very few optimization variables. Regarding point c), the fast computation times are partly due to the analytic propagation of the costates. Thus, it is desirable that the costates can be propagated analytically, for example with a state transition matrix. If not, the computation times would be slower, and the costates may become unstable from the numerical propagation.

Pontani's IHM and Stupik's framework for solving a differential game can be successfully applied to multiple types of zero-sum, pursuit-evasion games using the Tschauner-Hempel equations of motion and the Yamanaka-Ankersen state transition matrix. Thus, the developed method can be used for games with respect to a non-circular player or reference orbit, and the costates can still be propagated analytically. Pursuit-evasion games can be formulated and solved for not only the intercept case, but also for the following games: a) rendezvous, b) obtain Sun vector, c) match energy, d) obtain Sun vector and match energy, and e) match energy and remain close. The terminal constraints for these games can be formulated in such a way that the IHM can still be used, and such that the problem remains tractable. Also, the differential game solutions can be somewhat verified by solving multiple one-sided optimal control problems and comparing the solutions to the differential game solution.

The solutions to these games provide the open-loop strategies for the pursuer and evader and may help to answer several questions. First, the solutions may provide the worst-case time (and corresponding fuel) for the pursuer to accomplish its inspection

goal since it is based on the evader playing optimally. If the evader behaves any differently, then the pursuer may ideally be able to accomplish its inspection goal sooner and thus with less fuel. Second, the game solutions may provide the actual strategy for the evader to implement in such a scenario, no matter the strategy of the pursuer. If the evader uses the open-loop strategy generated from these solutions, then it may give it the best chance at evading the pursuer, forcing the pursuer to use the most time (and corresponding fuel) to chase the evader. It may force the pursuer to discontinue its pursuit, if the pursuer realizes the time and fuel which would be required continue pursuing the optimally evading evader. Third, these game solutions may aid in the design process. Based off the expected capabilities of the other player, the propulsion properties of the player of interest may be designed to perform to a certain level in the differential games.

# VI. Summary

Optimal, finite-thrust guidance methods have been successfully developed for various constrained proximity operations maneuvers of an inspector satellite maneuvering about a non-maneuvering or maneuvering RSO. As hypothesized in Chapter I, it has been shown that by using advanced optimization techniques such as pseudospectral methods and metaheuristic methods, time and fuel-optimal guidance can be found for complex, highly constrained, nonlinear problems. Three different finite-control types have been investigated, where the developed algorithms for Problems A, B, and C account for each control type. Both NMCs and teardrops were thoroughly investigated, and constraints such as keep-out zones, lighting constraints, and collision avoidance constraints were successfully incorporated into the problems. The following sections describe the contributions generated by answering each research question, and any related potential future work.

## 6.1 Contributions

The main contributions are algorithms which answer each research question. Along with each main contribution are some ways in which they were evaluated. The main contributions are:

1. A path planning algorithm using pseudospectral methods to determine minimum-time and minimum-fuel solutions for an inspector satellite with one, finite-thrust, body-fixed engine and maximum slew rates. Furthermore, an algorithm which uses this control type to inject an inspector satellite into an NMC about a non-maneuvering RSO, and transfer to an orthogonal one in order to reach viewing angles from all eight octants surrounding the RSO, while staying out of a keep-out cone.

This algorithm could be used to generate guidance for an initial inspection-type mission of an RSO. One of the ways this algorithm was evaluated was by comparing the guidance generated from the multi-phase problem to the guidance generated if the problems were optimized separately. It was shown that time and fuel could be saved by combining the two maneuvers into one, multi-phase optimization problem.

2. A path planning algorithm using various optimization techniques to determine minimum-time and minimum-fuel solutions for an inspector satellite with multiple on/off thrusters and capable of reorienting its thrust vector instantaneously. Furthermore, an algorithm which uses this control type to inject an inspector satellite into a relative teardrop trajectory (as well as into an NMC) about a non-maneuvering RSO, while adhering to lighting and collision constraints.

   This algorithm was evaluated several ways. One way was to compare the guidance generated with impulsive burns to the guidance generated where the finite nature of the burn was accounted for, and seeing that for many scenarios the impulsive burn solutions contained a significant amount of error. Another way was to use the control generated in an independent, high-fidelity propagator and seeing the the resulting trajectory was close enough to the desired trajectory.

3. A differential game solution technique using metaheuristic methods to determine optimal zero-sum, open-loop strategies for a pursuer and evader, both using constant, steerable thrust, where the pursuer wishes to achieve a specific inspection goal as soon as possible and the evader wishes to prolong that condition as long as possible, for various inspection goals.

   One of the ways this algorithm was evaluated was by ensuring that the

solutions obtained were the differential game solutions. This was done by running many one-sided optimal control problems with suboptimal strategies, and seeing that the differential game solution held.

4. A reliable application of metaheuristic optimization algorithms to provide initial guesses for and/or complete solutions to the problems addressed in Problems B and C.

   The reliability of these algorithms were evaluated in Problem B by running them many times, and determining how often they were an adequate initial guess for an NLP solver. For Problem C, they were run many times for the one-sided problems and always led to a solution for the cases run.

The solutions generated by these algorithms show that highly constrained and nonlinear problems can successfully be formulated and solved. They account for more realistic spacecraft control and constraints, and thus can generate more accurate guidance. This helps to save time and/or fuel, and if used on an actual satellite may help to improve a satellite's capability and/or mission life.

### 6.1.1 Problem A Contributions.

Specific contributions from Problem A include:

- A method to find optimal guidance (with respect to time or fuel) for a combined formation establishment *and* reconfiguration mission, given the minimum amount of time desired in the first NMC, which saves time and/or fuel compared to optimizing each mission component separately.

- A way to incorporate one, body-fixed engine with variable thrust and slew rate limits into an optimal control problem, thus coupling the translational trajectory to rotational constraints.

- A technique for augmenting the cost function such that results are relatively smooth and suitable for satellite operations.
- Constraint formulations which allow the optimizer to converge to one of two initial NMCs, and enter at any point along that NMC.
- A way to produce a family of optimal solutions via a homotopic approach, starting with the minimum-time solution.
- A conference paper [80] containing the research for Problem A: "Optimal Slew-Rate-Limited Guidance for Combined Formation Establishment and Reconfiguration of Inspector Satellite With Exclusion Cone".

### 6.1.2 Problem B Contributions.

Specific contributions from Problem B include:

- A parameterization of the control for sequences composed of constant acceleration magnitude and direction burn and coast phases, where the number of control variables resulting from the parameterization is relatively small. Also, the parameterization transforms the optimal control problem into a static optimization problem, without any discretization required. The control variables can be bounded in this static optimization problem, for minimum-fuel problems, such that the search space is not limited whatsoever.
- A semi-analytic guidance method for on/off thrusters, where the final HCW states after a defined sequence can be calculated analytically, given the control variables. This avoids the need to numerically propagate the HCW states when using a metaheuristic algorithm or an NLP solver, thus greatly increasing the speed of those algorithms.

- A burn-burn and a burn-coast-burn sequence used to find minimum-time and minimum-fuel solutions respectively.

- A coast-burn-coast-burn sequence where the first coast phase can save fuel compared to using just a burn-coast-burn sequence and provide the flexibility needed to satisfy certain constraints.

- A way to use the LROEs and Lovell's teardrop parameters to generate the desired NMC or teardrop, where the user only needs to choose four or five parameters to define the desired trajectory. Additionally, teardrop design clarifications for Lovell's teardrop parameters.

- A way to use $\beta$ as an optimization variable to allow the entry point to move along the trajectory within bounds on $\beta$ to further reduce the performance index.

- Reliable and fast initial guess methods: CW Targeting, MATLAB's PSO, modified MATLAB PSO, and MATLAB's GA. If MATLAB's Optimizaton Toolbox is available, use MATLAB's GA with the non-default *Penalty Algorithm*. If the toolbox is unavailable, use CW Targeting or the modified MATLAB PSO.

- Fast mid and high-fidelity models, where the high-fidelity model takes into account higher-fidelity equations of motion and continuous mass loss. The computation times are fast for the mid-fidelity model due to the low number of optimization variables, the analytic propagation of the sequence, and the good initial guesses, and the computation times are fast for the high-fidelity model due to the good initial guesses from the mid-fidelity model and the use of pseudospectral methods.

- Recommendations on NLP algorithms to use for best chances at convergence:

*sqp* for the tight sunlight constraint, and *interior-point* for the relaxed sunlight constraint. If both methods work well, then use *sqp* due to its faster computation times.

- Recommendation to use the relaxed sunlight constraint when possible since it converges more often than the tight sunlight constraint.

- Analytic derivatives for NLP solvers, where it has been shown that supplying exact, analytic first derivatives can increase chances of convergence. (Some NLP solvers also require user-supplied first derivatives, like IPOPT).

- Two types of sunlight constraints and field-of-view constraints for the Earth and the Moon, developed for both NMCs and teardrops, where these constraints are incorporated into the terminal constraints and are not path constraints.

- Collision avoidance methods, to include passive collision avoidance, using the developed algorithms.

- Methods to produce a range of optimal solutions with the developed algorithms.

- Visualization and validation performed with FreeFlyer.

- A conference paper [81] containing the research for Problem B-1: "Optimal Inspector Satellite Guidance to Quasi-Hover Via Relative Teardrop Trajectories".

- A journal article [82] containing the research for Problem B-1: "Optimal inspector satellite guidance to quasi-hover via relative teardrop trajectories".

- Two conference papers [83, 84] containing the research for Problem B-2: "Computationally Efficient Methods for Fuel Optimal Proximity Maneuvers with Constraints", and "Fuel Optimal, Finite Thrust Guidance Methods to Circumnavigate with Lighting Constraints".

- A conference paper [85] containing the research for Problem B-3: "Optimal Guidance for Relative Teardrops with Lighting and Collision Constraints".

### 6.1.3 Problem C Contributions.

Specific contributions from Problem C include:

- A way to use the Tschauner-Hempel equations of motion and the Yamanaka-Ankersen STM within Pontani's IHM and Stupik's differential game framework.
- Constraint formulations which can be used within the IHM and are linear with respect to the states such that the resulting boundary value problem is easier to solve. These constraints have been developed for the following games (in addition to intercept): rendezvous, obtain Sun vector, match energy, obtain Sun vector *and* match energy, and match energy and remain close.
- Relatively fast boundary value problem solution techniques by using the IHM and either MATLAB's PSO, or MATLAB's GA with the non-default *Penalty Algorithm*.
- A method using the IHM to help validate the differential game solutions by solving multiple one-sided optimal control problems using optimal or suboptimal strategies for the deterministic player.
- Worst case scenarios to expect on orbit or to aid in propulsion system design.
- Optimal evader strategies which could be used regardless of the pursuer strategies.
- Demonstration that the IHM is a suitable method for solving these types of differential games.

## 6.2 Future Work

Future work for both Problems A and B include experimenting with higher-fidelity equations of motion or propagators as the dynamics inside of GPOPS-II and targeting the corresponding higher-fidelity natural trajectories. This would lead to even more accurate guidance and would still benefit from the mid-fidelity solutions serving as the initial guesses. Additionally, a model predictive control technique could be developed for Problems A and B, where the optimal control could be regenerated at points along the path as the error or predicted error grows too large. This could serve as a type of feedback control technique or could accompany a closed-loop controller.

For Problem B specifically, more burns could be added to the sequence, where the solution from the previous sequence would serve as an initial guess for the next iteration of the sequence. For example, the coast-burn-coast-burn solution could serve as the initial guess for a coast-burn-coast-burn-coast-burn sequence. It could be investigated if more burns reduce the performance index even more. Obviously, it may be difficult to keep generating and using analytic derivatives, and thus derivative approximations would have to be used. Also, for Problem B, a multi-phase problem may be able to be solved by using an extended sequence and adding another set of terminal constraints as linkage constraints. If possible, a multi-phase problem could be investigated where the inspector satellite must visit two different RSOs, and thus two different natural trajectories would be targeted in the same optimization problem (similar to Problem A). Regarding the analytic solution to a constant acceleration magnitude, constant direction burn, it could be investigated as to if similar expressions can be developed for the Tschauner-Hempel equations of motion. With respect to collision avoidance, an adaptive method could be developed to more intelligently select the points at which to check for a collision, such that collisions aren't checked for at every point along the trajectory. And finally, regarding passive collision avoidance

specifically, GPOPS-II could be used, where the passive trajectory could be handled as a staging event, where the satellite splits into two, one having made the final burn, and one having failed to make the final burn.

For Problem C, a good study would be to determine the difference in the behavior of the evader between prolonging the pursuer's inspection goal as long as possible vs. actually trying to achieve the inspection goal itself. It would also be desirable to determine a way to weigh the fuel being used vs. achieving the goal in the game. That is, each player may want to try and win the game, but at what point is it no longer worth it, based on the fuel required to do so? And finally, with the current setup, a Monte Carlo analysis could be run, where the initial conditions and spacecraft properties vary, in order to determine relationships between those parameters and the outcome of the game.

While there is still much work to be done, the research herein establishes the necessary framework and shows the benefits of using optimization algorithms to solve the exhibited problems. The optimal guidance solutions generated will save more time and/or fuel and ultimately improve SSA of the GEO belt.

# Bibliography


1. T. A. Lovell and D. L. Brown, "Impulsive-Hover Satellite Trajectory Design for Rendezvous and Proximity Operation Missions," in *American Astronautical Society*, pp. 1–16, 2007.

2. US White House, "National Space Policy of the United States of America," 2010.

3. US Joint Chiefs of Staff, "Space Operations," 2013.

4. SAIC, "Space-Track," 2017.

5. G. H. McCall and J. H. Darrah, "Space Situational Awareness," *Air & Space Power Journal*, no. November-December, pp. 6–16, 2014.

6. AFSPC Fact Sheet, "Geosynchronous Space Situational Awareness Program (GSSAP)," 2015.

7. AFRL Fact Sheet, "Automated Navigation and Guidance Experiment for Local Space (ANGELS)," 2014.

8. K. T. Alfriend, S. R. Vadali, P. Gurfil, J. P. How, and L. S. Breger, *Spacecraft Formation Flying*. Elsevier, first ed., 2010.

9. J. Tschauner and P. Hempel, "Optimale Beschleunigeungsprogramme fur das Rendezvous-Manover," *Astronautica Acta*, vol. 10, pp. 296–307, 1964.

10. K. Yamanaka and F. Ankersen, "New State Transition Matrix for Relative Motion on an Arbitrary Elliptical Orbit," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 60–66, 2002.

11. P. Sengupta and S. R. Vadali, "Relative Motion and the Geometry of Formations in Keplerian Elliptic Orbits," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 953–964, 2007.

12. R. S. Wiltshire and W. H. Clohessy, "Terminal Guidance System for Satellite Rendezvous," *Journal of the Aerospace Sciences*, vol. 27, no. 9, pp. 653–658,674, 1960.

13. D. A. Vallado, *Fundamentals of Astrodynamics and Applications*. Hawthorne: Microcosm Press, third ed., 2007.

14. T. A. Lovell and S. Tragesser, "Guidance for Relative Motion of Low Earth Orbit Spacecraft Based on Relative Orbit Elements," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, (Providence), pp. 1–16, 2004.

15. T. A. Lovell and D. A. Spencer, "Relative Orbital Elements Formulation Based upon the Clohessy-Wiltshire Equations," *Journal of the Astronautical Sciences*, vol. 61, pp. 341–366, 2014.

16. C. Sabol, R. Burns, and C. A. McLaughlin, "Satellite Formation Flying Design and Evolution," *Journal of Spacecraft and Rockets*, vol. 38, no. 2, pp. 270–278, 2001.

17. T. A. Lovell and M. V. Tollefson, "Calculation of Impulsive Hovering Trajectories via Relative Orbit Elements," in *AAS/AIAA Astrodynamics Specialists Conference*, (Lake Tahoe), pp. 1–17, 2005.

18. A. S. Hope and A. J. Trask, "Pulsed-Thrust Method for Hover Formation Flying," in *AAS/AIAA Astrodynamics Specialists Conference*, (Big Sky, Montana), pp. Paper 03–655, 2003.

19. J. T. Betts, "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

20. B. A. Conway, "A Survey of Methods Available for the Numerical Optimization of Continuous Dynamic Systems," *Journal of Optimization Theory and Applications*, vol. 152, pp. 271–306, 2012.

21. A. V. Rao, "A Survey of Numerical Methods for Optimal Control," *Advances in the Astronautical Sciences*, 2009.

22. M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. Hoboken: John Wiley & Sons, Inc., third ed., 2006.

23. G. Elnagar, M. A. Kazemi, and M. Razzaghi, "The Pseudospectral Legendre Method for Discretizing Optimal Control Problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 10, pp. 1793–1796, 1995.

24. D. Benson, G. Huntington, T. Thorvaldsen, and A. Rao, "Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 6, pp. 1435–1440, 2006.

25. D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "A Unified Framework for the Numerical Solution of Optimal Control Problems Using Pseudospectral Methods," *Automatica*, vol. 46, no. 11, pp. 1843–1851, 2010.

26. D. Garg, M. A. Patterson, C. Francolin, C. L. Darby, G. T. Huntington, W. W. Hager, and A. V. Rao, "Direct Trajectory Optimization and Costate Estimation of Finite-Horizon and Infinite-Horizon Optimal Control Problems Using a Radau Pseudospectral Method," *Computational Optimization and Applications*, vol. 49, pp. 335–358, 2011.

27. M. A. Patterson and A. V. Rao, "GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp–Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," *ACM Transactions on Mathematical Software*, vol. 41, no. 1, pp. 1–37, 2014.

28. D. Garg, *Advances in Global Pseudospectral Methods for Optimal Control*. Dissertation, University of Florida, 2011.

29. P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Journal on Optimization*, vol. 12, no. 4, pp. 979–1006, 2002.

30. A. Wächter and L. T. Biegler, "On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

31. J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

32. M. Pontani and B. A. Conway, "Particle Swarm Optimization Applied to Space Trajectories," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 5, pp. 1429–1441, 2010.

33. M. Pontani and B. Conway, "Minimum-Fuel Finite-Thrust Relative Orbit Maneuvers via Indirect Heuristic Method," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 5, pp. 913–924, 2015.

34. J. T. Betts and W. P. Huffman, "Sparse Optimal Control Software (SOCS)," tech. rep., Boeing Information and Support Services, The Boeing Company, 1997.

35. C. L. Ranieri, "Path-Constrained Trajectory Optimization for Proximity Operations," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, (Honolulu), pp. 1–17, 2008.

36. G. T. Huntington, D. Benson, and A. V. Rao, "Optimal Configuration of Tetrahedral Spacecraft Formations," *The Journal of the Astronautical Sciences*, vol. 55, no. 2, pp. 141–169, 2007.

37. G. T. Huntington and A. V. Rao, "Optimal Reconfiguration of Spacecraft Formations Using the Gauss Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 3, pp. 689–698, 2008.

38. Y. Wu, X. Cao, Y. Xing, P. Zheng, and S. Zhang, "Relative Motion Integrated Coupled Control for Spacecraft Formation Using Gauss Pseudospectral Method," *Transactions of the Japan Society for Aeronautical and Space Sciences*, vol. 52, no. 177, pp. 125–134, 2009.

39. B. Wu, D. Wang, E. K. Poh, and G. Xu, "Nonlinear Optimization of Low-Thrust Trajectory for Satellite Formation: Legendre Pseudospectral Approach," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 4, pp. 1371–1381, 2009.

40. B.-L. Wu, D.-W. Wang, and E. K. Poh, "Energy-Optimal Low-Thrust Satellite Formation Manoeuvre in Presence of J2 Perturbation," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 225, pp. 961–968, 2011.

41. G. Ma, H. Huang, and Y. Zhuang, "Time Optimal Trajectory Planning for Reconfiguration of Satellite Formation with Collision Avoidance," in *IEEE International Conference on Control and Automation*, (Xiamen), pp. 476–479, 2010.

42. S. Lee and I. Hwang, "Hybrid High- and Low-Thrust Optimal Path Planning for Satellite Formation Flying," in *AIAA Guidance, Navigation, and Control Conference*, (Minneapolis), pp. 1–13, 2012.

43. R. Inampudi and H. Schaub, "Optimal Reconfigurations of Two-Craft Coulomb Formation in Circular Orbits," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 6, pp. 1805–1815, 2012.

44. X. Huang, Y. Yan, and Y. Zhou, "Optimal Spacecraft Formation Establishment and Reconfiguration Propelled by the Geomagnetic Lorentz Force," *Advances in Space Research*, vol. 54, pp. 2318–2335, 2014.

45. X. Huang, Y. Yan, Y. Zhou, and H. Zhang, "Pseudospectral Method for Optimal Propellantless Rendezvous Using Geomagnetic Lorentz Force," *Applied Mathematics and Mechanics*, vol. 36, no. 5, pp. 609–618, 2015.

46. X. Zengwen, S. Peng, and Z. Yushan, "Optimal Reconfiguration Control of Electromagnetic Spacecraft Formation Using Gauss Pseudospectral Method," in *2015 2nd International Conference on Information Science and Control Engineering*, pp. 862–866, 2015.

47. J. Li, "Fuel-Optimal Low-Thrust Formation Reconfiguration Via Radau Pseudospectral Method," *Advances in Space Research*, vol. 58, pp. 1–16, 2016.

48. R. Bevilacqua and T. A. Lovell, "Analytical guidance for spacecraft relative motion under constant thrust using relative orbit elements," *Acta Astronautica*, vol. 102, pp. 47–61, 2014.

49. D. J. Irvin, R. G. Cobb, and T. A. Lovell, "Fuel-Optimal Maneuvers for Constrained Relative Satellite Orbits," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 3, pp. 960–973, 2009.

50. D. J. Irvin and R. G. Cobb, "Feasibility Study and Fuel Optimal Control Techniques for Constrained Relative Satellite Orbit Geometries," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, (Keystone, Colorado), pp. 1–9, 2006.

51. J. L. Williams and E. G. Lightsey, "Optimal Impulsive Maneuvering Within a Confined Hover Region," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, (Honolulu), pp. 1–12, AIAA, 2008.

52. J. Zhang, S. Zhao, and Y. Yang, "Characteristic Analysis for Elliptical Orbit Hovering Based on Relative Dynamics," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 4, pp. 2742–2750, 2013.

53. G. Deaconu, C. Louembet, and A. Théron, "Designing Continuously Constrained Spacecraft Relative Trajectories for Proximity Operations," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 7, pp. 1208–1217, 2015.

54. F. J. Franquiz, J. D. Muñoz, B. Udrea, and M. J. Balas, "Optimal range observability maneuvers of a spacecraft formation using angles-only navigation," *Acta Astronautica*, 2018.

55. G. R. Frey, C. D. Petersen, F. A. Leve, I. V. Kolmanovsky, and A. R. Girard, "Constrained Spacecraft Relative Motion Planning Exploiting Periodic Natural Motion Trajectories and Invariance," *Journal of Guidance, Control, and Dynamics*, 2017.

56. M. Pontani, P. Ghosh, and B. A. Conway, "Particle Swarm Optimization of Multiple-Burn Rendezvous Trajectories," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 4, pp. 1192–1207, 2012.

57. D. J. Showalter and J. T. Black, "Responsive Theater Maneuvers via Particle Swarm Optimization," *Journal of Spacecraft and Rockets*, vol. 51, no. 6, pp. 1976–1985, 2014.

58. D. J. Showalter and J. T. Black, "Optimal Continuous-Thrust Responsive Theater Maneuvers," *Journal of Spacecraft and Rockets*, vol. 52, no. 5, pp. 1375–1387, 2015.

59. D. J. Showalter and J. T. Black, "Near-Optimal Geostationary Transfer Maneuvers with Cooperative En-Route Inspection Using Hybrid Optimal Control," *Acta Astronautica*, vol. 105, pp. 395–406, 2014.

60. H. Huang, Y. Zhuang, G. Ma, and Y. Lv, "Optimal Spacecraft Formation Reconfiguration with Collision Avoidance Using Particle Swarm Optimization," *Information Technology And Control*, vol. 41, no. 2, pp. 143–150, 2012.

61. H. Huang and Y. Zhuang, "Optimal Satellite Formation Reconfiguration Using Co-Evolutionary Particle Swarm Optimization in Deep Space," *Acta Astronautica*, vol. 113, pp. 149–163, 2015.

62. M. Pontani and B. A. Conway, "Optimal Finite-Thrust Rendezvous Trajectories Found via Particle Swarm Algorithm," *Journal of Spacecraft and Rockets*, vol. 50, no. 6, pp. 1222–1234, 2013.

63. M. Pontani, "Symmetry Properties of Optimal Relative Orbit Trajectories," *Mathematical Problems in Engineering*, pp. 1–11, 2015.

64. R. Isaacs, *Differential Games*. 1965.

65. A. E. Bryson and Y.-C. Ho, *Applied Optimal Control: Optimization, Estimation, and Control*. Washington D.C: Hemisphere Publishing Corporation, 1975.

66. K. Horie and B. A. Conway, "Optimal Fighter Pursuit-Evasion Maneuvers Found via Two-Sided Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 1, pp. 105–112, 2006.

67. K. Horie and B. A. Conway, "Genetic Algorithm Preprocessing for Numerical Solution of Differential Games Problems," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 6, pp. 1075–1078, 2004.

68. M. Pontani and B. A. Conway, "Numerical Solution of the Three-Dimensional Orbital Pursuit-Evasion Game," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 2, pp. 474–487, 2009.

69. D. Shen, K. Pham, E. Blasch, H. Chen, and G. Chen, "Pursuit-evasion orbital game for satellite interception and collision avoidance," 2011.

70. E. P. Blasch, K. Pham, and D. Shen, "Orbital satellite pursuit-evasion game-theoretical control," in *11th International Conference on Information Science, Signal Processing and their Applications, (ISSPA)*, pp. 1007–1012, 2012.

71. D. Shen, B. Jia, G. Chen, K. Pham, and E. Blasch, "Space based sensor management strategies based on informational uncertainty pursuit-evasion games," in *IEEE National Aerospace and Electronics Conference (NAECON)*, pp. 95–101, jun 2015.

72. D. Shen, B. Jia, G. Chen, E. Blasch, and K. Pham, "Pursuit-evasion games with information uncertainties for elusive orbital maneuver and space object tracking," 2015.

73. D. Shen, B. Jia, G. Chen, K. Pham, and E. Blasch, "Pursuit-evasion game theoretic uncertainty oriented sensor management for elusive space objects," in *IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS)*, pp. 156–163, jul 2016.

74. J. Stupik, M. Pontani, and B. Conway, "Optimal Pursuit/Evasion Spacecraft Trajectories in the Hill Reference Frame," in *AIAA/AAS Astrodynamics Specialist Conference*, (Minneapolis), pp. 1–15, 2012.

75. J. M. Stupik and B. Conway, *Optimal Pursuit/Evasion Spacecraft Trajectories in the Hill Reference Frame*. Thesis, University of Illinois at Urbana-Champaign, 2013.

76. S. Sun, Q. Zhang, R. Loxton, and B. Li, "Numerical Solution of a Pursuit-Evasion Differential Game Involving Two Spacecraft in Low Earth Orbit," *Journal of Industrial and Management Optimization*, vol. 11, no. 4, 2015.

77. J. Selvakumar and E. Bakolas, "A Pursuit-Evasion Game in the Orbital Plane," in *AAS/AIAA Spaceflight Mechanics Meeting*, (San Antonio), 2017.

78. S. Hernandez and M. R. Akella, "Lyapunov-based guidance for orbit transfers and rendezvous in Levi-Civita coordinates," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 4, pp. 1170–1181, 2014.

79. S. Gutman, M. Esh, and M. Gefen, "Simple linear pursuit-evasion games," *Computers & Mathematics with Applications*, vol. 13, no. 1-3, pp. 83–95, 1987.

80. E. R. Prince, R. G. Cobb, and J. A. Hess, "Optimal Slew-Rate-Limited Guidance for Combined Formation Establishment and Reconfiguration of Inspector Satellite With Exclusion Cone," in *27th AAS/AIAA Space Flight Mechanics Meeting*, (San Antonio), 2017.

81. E. R. Prince and R. G. Cobb, "Optimal Inspector Satellite Guidance To Quasi-Hover Via Relative Teardrop Trajectories," in *International Workshop on Spacecraft Constellations and Formation Flight*, (Boulder, CO), pp. 1–20, 2017.

82. E. R. Prince and R. G. Cobb, "Optimal inspector satellite guidance to quasi-hover via relative teardrop trajectories," *Acta Astronautica*, 2017.

83. E. R. Prince, R. W. Carr, and R. G. Cobb, "Computationally Efficient Methods for Fuel Optimal Proximity Maneuvers with Constraints," in *AAS/AIAA Astrodynamics Specialist Conference*, (Stevenson, WA), 2017.

84. E. R. Prince, R. W. Carr, and R. G. Cobb, "Fuel Optimal, Finite Thrust Guidance Methods to Circumnavigate with Lighting Constraints," in *Advanced Maui Optimal and Space Surveillance and Technologies Conference*, (Maui), 2017.

85. E. R. Prince and R. G. Cobb, "Optimal Guidance for Relative Teardrops with Lighting and Collision Constraints," in *AIAA Guidance, Navigation, and Control Conference*, (Kissimmee, Florida), 2018.

86. D. E. Kirk, *Optimal Control Theory: An Introduction*. Mineola: Dover Publications, Inc., 1970.

87. H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*. Reston: American Institute of Aeronautics and Astronautics, 2003.

88. J. Li and X.-n. Xi, "Fuel-Optimal Low-Thrust Reconfiguration of Formation-Flying Satellites via Homotopic Approach," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 6, pp. 1709–1717, 2012.

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704–0188*



| 1. REPORT DATE *(DD–MM–YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From — To)* |
|---|---|---|
| 16–08–2018 | Doctoral Dissertation | Sept 2015 — Sept 2018 |

**4. TITLE AND SUBTITLE**

OPTIMAL FINITE THRUST GUIDANCE METHODS FOR CONSTRAINED SATELLITE PROXIMITY OPERATIONS INSPECTION MANEUVERS

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**

Prince, Eric R., Capt, USAF

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

**8. PERFORMING ORGANIZATION REPORT NUMBER**

AFIT-ENY-DS-18-S-071

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

Intentionally Left Blank

**10. SPONSOR/MONITOR'S ACRONYM(S)**

N/A

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

**12. DISTRIBUTION / AVAILABILITY STATEMENT**

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**13. SUPPLEMENTARY NOTES**



**14. ABSTRACT**

Algorithms are developed to find optimal guidance for an inspector satellite operating nearby a resident space object (RSO). For a non-maneuvering RSO, methods are first developed for a satellite subject to maximum slew rates to conduct an initial inspection of an RSO, where the control variables include the throttle level and direction of the thrust. Second, methods are developed to optimally maneuver a satellite with on/off thrusters into a natural motion circumnavigation or teardrop trajectory, subject to lighting and collision constraints. It is shown that for on/off thrusters, a control sequence can be parameterized to a relatively small amount of control variables and the relative states can be analytically propagated as a function of those control variables. For a maneuvering RSO, differential games are formulated and solved for an inspector satellite to achieve multiple inspection goals, such as aligning with the Sun vector or matching the RSO's energy. The developed algorithms lead to fuel and time savings which can increase the mission life and capabilities of inspector satellites and thus improve space situational awareness for the U.S. Air Force.

**15. SUBJECT TERMS**

Satellite rendezvous and proximity operations (RPO), optimal control, trajectory optimization, path planning, differential games

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Dr. Richard G. Cobb, AFIT/ENY |
| U | U | U | U | 249 | 19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636, x4559; richard.cobb@afit.edu |