3-2020

Linear Regression Models Applied to Imperfect Information Spacecraft Pursuit-evasion Differential Games

Dax Linville

LINEAR REGRESSION MODELS APPLIED TO IMPERFECT INFORMATION SPACECRAFT PURSUIT-EVASION DIFFERENTIAL GAMES

THESIS

Dax Linville, Capt, USAF

AFIT-ENY-MS-20-269

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This is an academic work and should not be used to imply or infer actual mission capability or limitations.

AFIT-ENY-MS-20-269

LINEAR REGRESSION MODELS APPLIED TO IMPERFECT INFORMATION SPACECRAFT PURSUIT-EVASION DIFFERENTIAL GAMES

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science (Astronautical Engineering)

Dax Linville, BS

Capt, USAF

March 21, 2020

AFIT-ENY-MS-20-269

LINEAR REGRESSION MODELS APPLIED TO IMPERFECT INFORMATION SPACECRAFT PURSUIT-EVASION DIFFERENTIAL GAMES

THESIS

Dax Linville, BS
Capt, USAF

Committee Membership:

Maj Joshuah A. Hess, PhD Chair	Date
Richard G. Cobb, PhD Member	Date
Maj Costantinos Zagaris, PhD Member	Date

Abstract

Within satellite rendezvous and proximity operations lies pursuit-evasion differential games between two spacecraft. For example, a rescuing satellite pursuing an uncontrolled "evading" satellite can be formulated as a pursuit-evasion differential game. The extent of possible outcomes resulting from the interaction of the pursuer and evader satellites can be mathematically bounded by pursuit-evasion differential games where each player employs optimal strategies against the other. Typically, pursuit-evasion games are formulated as optimal control problems that require significant computational expense and time to solve. A linear regression model is developed to enable on-board computation of these pursuit-evasion games for possible autonomous action. Built from a large data set of optimal control solutions, the model is solved at geosynchronous altitudes using an indirect heuristic method with particle swarm optimization (IHM-PSO). Using linearized dynamics, the model is shown to quickly map pursuer relative starting positions to final capture positions and estimate capture time. The model is 3.8 times faster than the IHM-PSO method for arbitrary performance scaling between the two spacecraft and arbitrary pursuer starting positions on an initial relative orbit about the evader (at the origin of the coordinate frame). A parameter study is performed to analyze the effect of uncertainty in the pursuer starting position, and to develop "rules of thumb" trends. The solution trajectories are validated in a closed-loop controller in order to ensure their viability. The trajectories are also propagated using higher fidelity simulation software, showcasing how the developed model is accurate enough for real-world mission planning. Thus the linear regression model is shown to be well suited for onboard implementation capable of evaluating results of optimal control pursuit-evasion differential games for optimal response and autonomous mission planning.

Life is meta-heuristical optimization. How do you know what's good unless you have lots of data from past experiences? Can life then be described mathematically?

Acknowledgements

First and foremost, I would like to thank my advisor, Maj Joshuah Hess, for leading me on an academic journey to last a lifetime. Thank you for your patience, and thoughtful instruction over the 18 months. I would also like to thank Dr. Cobb and Maj Zagaris for both agreeing to be on my committee, as well as for their thoughtful advice and feedback. Both of their instruction has helped shape my controls knowledge immensely. Also, thank you Dr. Alan Lovell and Dr. Andy Sinclair at the Air Force Research Laboratory, Space Vehicles Directorate, for your sponsorship of this work. And finally, many thanks to my wife for her endless support throughout this AFIT endeavor. This thesis would largely not be what it is without her support in helping raise two young children and allowing me to find a work-life balance.

Dax Linville

Table of Contents

Page

List of Figures

Figure Page

Figure Page

List of Tables

List of Abbreviations

ECI	Earth-Centered Inertial
EKF	Extended Kalman Filter
GEO	Geosynchronous
GNC	Guidance, Navigation, and Control
HCW	Hill-Clohessy-Wiltshire
HPOP	High-Precision Orbit Propagator
IHM	Indirect Heuristic Method
LKF	Linear Kalman Filter
LQR	Linear Quadratic Regulator
LVLH	Local Vertical Local Horizontal
NERM	Nonlinear Equations of Relative Motion
NMC	Natural Motion Circumnavigation
PSO	Particle Swarm Optimization
ROE	Relative Orbital Elements
RPO	Rendezvous & Proximity Operations
STM	State Transition Matrix
TH	Tschauner-Hempel Equations
UKF	Unscented Kalman Filter

List of Symbols

μ	Earth standard gravitational parameter
α	control azimuth angle for thruster
ϕ	control elevation angle for thruster
$\bar{\lambda}$	costate vector
ν_r	relative orbit phasing angle
$\bar{\psi}$	terminal constraint vector
ν_i	unknown values for terminal costates
$[A]$	plant matrix
$[B]$	control influence matrix
a_c	chief reference orbit semi-major axis
a_r	relative orbit ellipse semi-major axis
e	chief reference orbit eccentricity
f	true anomaly
$\mathcal{H}$	Hamiltonian
J	cost functional
$\mathcal{L}$	running cost
n	mean motion of chief reference orbit
t	time
$\bar{u}$	control vector
x	radial component of relative position as a function of time
y	in-track component of relative position as a function of time
z	cross-track component of relative position as a function of time

LINEAR REGRESSION MODELS APPLIED TO IMPERFECT INFORMATION SPACECRAFT PURSUIT-EVASION DIFFERENTIAL GAMES

I. Introduction

1.1 Motivation

Pursuit-evasion games can be played in the space environment, where two agents are involved in a game of "cat and mouse." For example, the pursuer agent could be a rescuing satellite that is trying to rendezvous with an evader agent, who could be a satellite in need of servicing. Prior to spending time computing a rendezvous trajectory, the pursuer spacecraft needs to make the decision whether or not to chase the evading spacecraft. Assuming capture is possible, the pursuer needs to characterize the game, (prior to playing it) in order to determine the time and travel distance required. The motivation for this research is to provide the ability for spacecraft to autonomously characterize an imminent pursuit-evasion game by developing a new algorithm for rapid estimation of optimal capture time and position. In order for this algorithm to be effective, the solution process must be accurate enough for on-orbit use, fast enough for real-time actions to be made from the information provided, and require minimal computational power in order to fit within the confines of typical space-based hardware. If the pursuer and evader employ optimal strategies, then the solution can be obtained by solving an optimal control problem. Unfortunately, optimal control problems are not typically computed on-board spacecraft (except in certain cases like model predictive control) due to both the high computational power required, and minutes of computation time typically needed to solve them. This re-

search seeks to eliminate these problems by pre-evaluating a multitude of solutions, and then reducing the data set through the use of linear regression techniques in order to develop input-output relations. Instead of running an optimal control algorithm on board a spacecraft, one can instead evaluate a set of functions to obtain the predicted end state and final time with the same accuracy and in a fraction of the time. The output of the linear regression model could then be used as the initial guess to other optimization formulations if required. Finally, the linear regression model computational speed advantage over classical optimal control solution strategies lends itself well to large scale parameter studies. These parameter studies can be performed in order to analyze trends inherent within the total solution space. The identified trends can then be extracted to provide physical intuitiveness and "rules of thumb" for space-based pursuit-evasion games. For example, knowledge of these trends can help future mission planners predict the end state of pursuit-evasion games without employing computer software! The trends also serve the purpose of informing Techniques, Tactics, and Procedures (TTPs) for space engagements that should be developed as the United States transitions from space as a peaceful domain into a wartime domain.

1.2 Research Overview

This research will seek to expand the knowledge pertaining to the treatment of mainly open-loop space-based pursuit-evasion differential games. To build a linear regression model, a collection of deterministic optimal control solutions will be built in layers that together will form a means from which to derive an input-output relationship. In particular, the pursuer initial states will be mapped to the final capture position and final capture time. With the linear regression models created, a parameter study will be conducted to uncover trends within the solution space for a gener-

alized pursuit-evasion game. In this parameter study a normally distributed number of sample points will be taken from an initial covariance ellipse that represents the uncertainty in the pursuit starting position from the perspective of the evader. Each pursuer starting position will be mapped to its respective capture position using the linear regression model. A new covariance ellipse will be calculated from the resulting group of capture points. In this way, a large amount of deterministic simulations can be brought together to model a stochastic effect. Finally, the trends that are shown as a result of the parameter study, as well as the solution trajectories, are validated for closed-loop performance and propagated in a high fidelity simulation environment to evaluate their applicability to real-world operations.

1.2.1 Problem Statement and Hypothesis.

There is a need to develop existing optimal control algorithms for implementation onboard spacecraft in preparation for spacecraft autonomy. It is hypothesized that linear regression models can be used to largely eliminate the roadblocks that keep optimal control solutions from being computed onboard spacecraft by improving the computational speed, lessening the memory burden, and reducing the required computational power.

1.2.2 Questions.

In this research effort the following research questions are posed:

1. Assuming that capture is possible, can a linear regression model be used to map initial pursuer positions to final capture positions?

2. Is the accuracy of the linear regression model sufficient enough to replace an optimal control solver?

3. Can the speed of the linear regression model make large scale parameter studies feasible without super-computing resources?

4. Are the generated optimal control solutions viable when applied in a Guidance Navigation and Control (GNC) closed-loop system?

5. Can linearized dynamics be used to model the nonlinear real-world orbital environment with acceptable levels of error when compared to higher fidelity propagators? Is this impact negligible enough for real-world operational use?

1.2.3 Tasks.

This research effort intends to explore the following:

1. Utilize existing two-player pursuit-evasion differential game solution algorithms to collect data for the linear regression model.

2. Build a linear regression model capable of predicting capture time and position for an arbitrary set of initial conditions.

3. Use the linear regression model to evaluate a large-scale parameter study to develop "rules of thumb" for two-player differential pursuit-evasion games.

4. Characterize the trends associated in the parameter study to inform future mission planning and create TTPs for orbital engagements.

5. Implement a closed-loop GNC system to track provided guidance trajectories in the presence of additive white Gaussian noise corrupting the state measurements.

6. Validate the linear regression model's accuracy against higher fidelity commercial propagators.

1.2.4 Scope.

This work is based solely on simulation with no experimental investigation. The dynamics models are entirely deterministic. This research effort studies a geostationary orbit (GSO) altitude, where each satellite remains sufficiently close to the origin of the coordinate frame (< 200 km), and will have sufficiently short games compared to the orbital period (< 3 hours). These assumptions allow the use of linearized dynamics, which will neglect environmental disturbances such as the Earth's gravitational J_2 effect, solar radiation pressure, and gravitational disturbances from other celestial bodies (like the Sun and Moon) effects. Furthermore, each spacecraft will be assumed to be 180 kg and carry one single low thrust continuous electric engine. For the closed-loop analysis, uncertainty in the propagated state will be incorporated by injecting additive white Gaussian noise into the deterministic dynamics model output.

1.3 Mathematical Notation

The following notation shown in Table 3 will be used in this thesis:

Table 3. Notation Used in Thesis

Quantity	Representation
Scalar	a
Vector	$\bar{a}$
Unit vector	$\hat{a}$
Magnitude of the vector	$\|\|\bar{a}\|\|$, $\|\|\hat{a}\|\|$
Matrix	[A]
Time derivative	$\dot{a}$
Derivative with respect to true anomaly	a'
Partial derivative of a with respect to x	$\frac{\partial a}{\partial x}$

1.4 Problem Introduction

The research questions identified in Section 1.2.2 are answered by four problems within this work. Problem A builds a 2D linear regression model to map pursuer initial position to final capture position for arbitrary initial conditions assuming a game played entirely within the orbital plane (no cross-track motion). An evader performance scale factor (SF) is introduced to vary the relative performance between the pursuer and evader in order to fully scope the solution space. Problem B expands upon Problem A by creating a 3D linear regression model, including cross-track motion. In essence, problem A solves a simplified solution space that is important in developing the framework for the more complicated Problem B. Problem C takes the two developed linear regression models and utilizes them in a large scale parameter study. The parameter study of problem C aims at identifying trends that can be used to create TTPs and inform future mission planning. Finally, problem D aims to address the viability of the generated solution trajectories used to build the linear regression models. Additionally problem D validates that the trends identified in problem C also exist in higher fidelity dynamics, thus making the problem C's "rules of thumb" relevant and useful for real-world mission planning. Problem D validates the use of simplified dynamics by showing similar results can be obtained with higher fidelity simulation software, namely AGI's System Tool Kit (STK). To further address the viability of the generated solutions, the solution trajectories are used as a reference within a developed closed-loop GNC system. Performing closed-loop analysis enables the study of effects such as: control saturation, use of simplified (mismatched) dynamics, and incorporation of measurement uncertainty. The four research problems tackled by this thesis address the research questions as shown in Table 4.

Table 4. Research Problems Traced to Research Questions

Research Question	PBM A	PBM B	PBM C	PBM D
1	x	x		
2				x
3			x	
4				x
5				x

1.5 Thesis Overview

The remainder of this document is comprised of four more sections. Chapter 2 presents a background and literature review of topics required for Chapter 3; the methodology, which is undertaken to arrive at Chapter 4, the results produced by this research effort. Finally, closing thoughts and ideas for future work are presented in Chapter 5.

II. Background

This chapter provides an introduction to pursuit-evasion differential games by first introducing the dynamical models for relative satellite motion, optimal control techniques, and finally methodologies for solving differential games. In addition, a primer on linear regression will be covered. Finally, state estimation theory and closed-loop control techniques are used within the GNC architecture for the closed-loop analysis, and thus is briefly covered.

2.1 Dynamics Models

2.1.1 Coordinate Systems.

The Earth-Centered-Inertial (ECI) coordinate system (sometimes referred to as the geocentric-equatorial coordinate system [1]) is shown in Figure 1. The ECI coordinate system's origin is coincident with the Earth's center of mass with its $\hat{x}$ axis pointing in the direction of Vernal Equinox. The $\hat{z}$ direction is aligned with the true north pole, and the $\hat{y}$ direction completes the right-handed, orthogonal coordinate system. While the ECI reference frame is not strictly inertial, it is often deemed "sufficiently inertial" for the purposes of most astrodynamics research within Earth's gravity well [2]. The ECI coordinate frame will be utilized when expressing the inertial satellite motion, and is the frame used to calculate time derivatives [2].

The Local Vertical, Local Horizontal (LVLH) coordinate frame, shown in Figure 2, will be utilized for relative satellite motion. The LVLH frame is a body fixed coordinate system that has the primary $\hat{x}$ axis aligned along the radius vector of the satellite's instantaneous position from the Earth's center. This is known as the radial component of the LVLH frame. The LVLH $\hat{z}$ axis, usually referred to as the cross-track component, is perpendicular to the orbital plane, pointing in the direction of

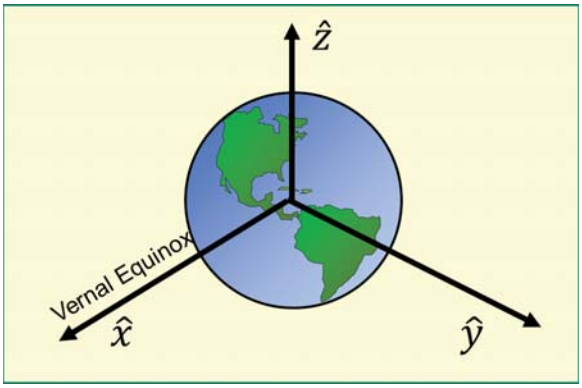


Figure 1. The Earth Centered Inertial (ECI) reference frame, adapted from [2]

the orbit's angular momentum vector ($\bar{h}$). The $\hat{y}$ axis, called the in track component, completes the right-handed orthonormal set. In the specific case of a circular orbit, which will be studied in this research effort, the $\hat{y}$ axis points in the direction of travel, along the velocity vector.

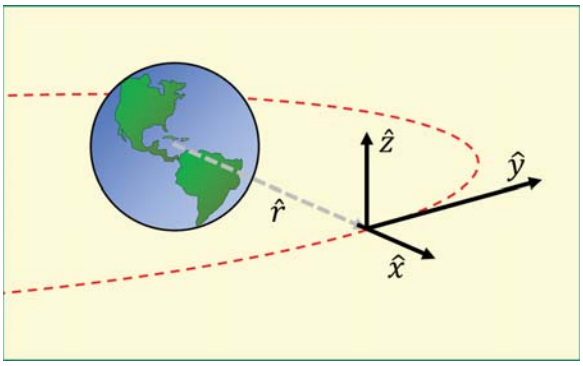


Figure 2. The Local Vertical, Local Horizontal (LVLH) Reference Frame, adapted from [2]

2.1.2 Inertial Dynamics Models.

The mathematical formulation to describe the realistic motion of a satellite through space can be difficult, or sometimes insurmountable. For example, Conway [3] notes that there are nonlinearities that arise from trigonometric functions, constant changes in system properties as on-board resources are used, and time varying forces such as

drag. The contribution of these nonlinear factors formulate a problem in which the boundary conditions are either not fully known, or depend upon the value being optimized [3]. Trying to fully describe realistic, perturbative orbital motion analytically is often futile; thus, simplifying assumptions and various numerical formulations have been derived, each with their own pros and cons. In the case of this research, orbital perturbations will be neglected for simplicity in the development of the linear regression model. Furthermore the nonlinear equations will be linearized in order to gain closed-form solutions that enable relatively quick computation of pursuit-evasion games. The impact of neglecting orbital perturbations, and use of linearized dynamics, will be analyzed to ensure accuracy is still acceptable for real-world use. Thus, the fundamental two-body problem will first be introduced, which leads to the full nonlinear equations of motion for unperturbed relative satellite motion. The results of reducing the full nonlinear equations into two different sets of linearized dynamics is shown in order to present a formulation for the dynamics underpinning the pursuit-evasion game.

2.1.2.1 The Two Body Problem.

The two body problem is the fundamental model that describes the gravitational pull that two bodies exert on each other based upon their mass and position vector ($\bar{r}$). The two bodies are usually assumed to be point masses, where one is the Earth's mass (M) and the other is the satellite's mass (m). Using Newton's laws the differential equation for satellite motion, with respect to the inertial frame, is [1]:

$$\ddot{\bar{r}} = -\frac{G(M+m)}{\|r\|^3}\bar{r} \simeq -\frac{\mu\bar{r}}{\|r\|^3} \tag{2.1}$$

Often, the mass of the satellite is considered negligible compared to the mass of the Earth. Thus, the gravitational constant (G) is multiplied by the Earth's mass (M),

and denoted by $\mu = 398601.2\ \mathrm{km}^3/\mathrm{s}^2$. The magnitude of the orbit radius can be found using known orbital elements [1]:

$$\|\bar{r}\| = \frac{a(1-e^2)}{1+e\cos f} \tag{2.2}$$

where a is the semi-major axis of the orbit, e is the eccentricity, and f is the true anomaly.

2.1.3 Relative Orbital Dynamics.

Thus far, orbital dynamics has been used to describe satellite motion with respect to the Earth. In order to study the relative motion of two satellites in a more intuitive sense, one satellite's motion will be described relative to the other. In this context, the reference satellite is usually called the chief, or leader. The other satellite is often referred to as the deputy, or follower [2]. By differencing the inertial position vectors, Equation (2.1), of both satellites, the separation vector $\bar{\rho}$ can be found:

$$\bar{\rho} = \bar{r}_d - \bar{r}_c \tag{2.3}$$

where subscripts to denote $\bar{r}_c$ and $\bar{r}_d$ for the chief and deputy respectively. Noting that $\bar{\rho}$ is still in the inertial reference frame, inertial time derivatives can be taken to yield:

$$\ddot{\bar{\rho}} = \ddot{\bar{r}}_d - \ddot{\bar{r}}_c \tag{2.4}$$

Substituting in the fundamental two body equation for motion (Equation (2.1)) for

each satellite and recognizing $\bar{r}_d = \bar{\rho} + \bar{r}_c$ yields:

$$\begin{aligned} \ddot{\bar{\rho}} &= \frac{-\mu \bar{r}_d}{\|r_d\|^3} - \left(\frac{-\mu \bar{r}_c}{\|r_c\|^3} \right) \\ &= \frac{-\mu(\bar{\rho} + \bar{r}_c)}{\|\bar{\rho} + \bar{r}_c\|^3} + \left(\frac{\mu \bar{r}_c}{\|r_c\|^3} \right) \end{aligned} \tag{2.5}$$

In this form, the dynamics are realized in the ECI reference frame and thus the axes do not rotate with the chief satellite along the orbital trajectory. In order to gain a more intuitive sense of motion, Equation (2.5) needs to be resolved from the inertial ECI reference frame to the rotating LVLH reference frame. Figure 3 shows the LVLH reference frame in red, and the ECI reference frame in black for reference.

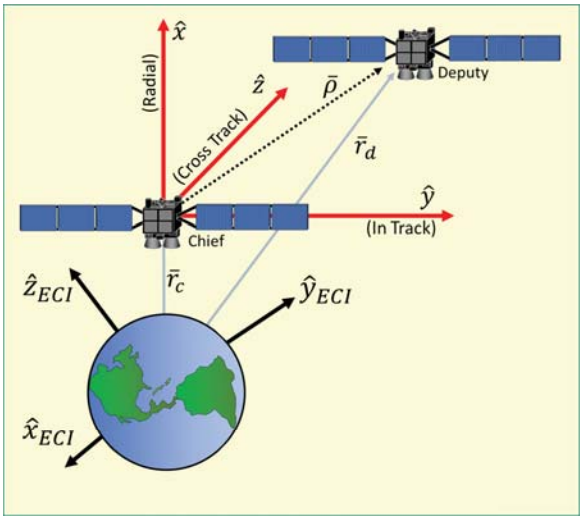


Figure 3. A Illustration of Relative Satellite Motion, adapted from [2]

Vectors in the ECI reference frame will be resolved in the LVLH reference frame by utilizing the rotation matrix $[R^{\mathcal{LI}}]$, which is built from the unit vectors [2]:

$$\begin{aligned} \hat{o}_r &= \frac{\bar{r}}{\|\bar{r}\|} \\ \hat{o}_h &= \frac{\bar{h}}{\|\bar{h}\|} = \frac{\bar{r} \times \bar{v}}{\|\bar{r} \times \bar{v}\|} \\ \hat{o}_\theta &= \hat{o}_h \times \hat{o}_r \end{aligned} \tag{2.6}$$

These unit vectors form the $[R^{\mathcal{L}I}]$ matrix [2]:

$$\begin{aligned}[R^{\mathcal{I}\mathcal{L}}] &= \begin{bmatrix} \hat{o}_r & \hat{o}_\theta & \hat{o}_h \end{bmatrix} \\ [R^{\mathcal{L}\mathcal{I}}] &= [R^{\mathcal{I}\mathcal{L}}]^T \end{aligned} \tag{2.7}$$

Using the rotation matrix $[R^{\mathcal{L}\mathcal{I}}]$ on Equation (2.3) yields:

$$\bar{\rho}^{\mathcal{L}} = \begin{bmatrix} R^{\mathcal{L}\mathcal{I}} \end{bmatrix} \bar{\rho}^{\mathcal{I}} \tag{2.8}$$

Next, using the transport theorem [4] twice and rearranging terms yields the resulting derivative of $\ddot{\bar{\rho}}$ with respect to the rotating LVLH reference frame given by Equation (2.9). Here superscripts on the left identify frames that derivatives are taken in and superscripts on the right denote the coordinate frame the vector is resolved in. In this manner, the inertial derivatives can be related to vectors resolved in a non-inertial frame:

$${}^{\mathcal{I}}\ddot{\bar{\rho}} = {}^{\mathcal{L}}\ddot{\bar{\rho}} + 2\big(\bar{\omega}_{\mathcal{L}\mathcal{I}} \times \bar{\rho}^{\mathcal{L}}\big) + \dot{\bar{\omega}}_{\mathcal{L}\mathcal{I}} \times \bar{\rho}^{\mathcal{L}} + \bar{\omega}_{\mathcal{L}\mathcal{I}} \times \big(\bar{\omega}_{\mathcal{L}\mathcal{I}} \times \bar{\rho}^{\mathcal{L}}\big) \tag{2.9}$$

where

$$\begin{aligned} \bar{\omega}_{\mathcal{L}\mathcal{I}} &= \begin{bmatrix} 0 & 0 & \dot{f} \end{bmatrix}^T \\ \bar{\rho}^{\mathcal{L}} &= \begin{bmatrix} x & y & z \end{bmatrix}^T \end{aligned}$$

and $\dot{f}$ is the time rate of change of the true anomaly, (x, y, z) are the radial, in-track and cross-track coordinates expressed in the LVLH relative frame. In this formulation, the pre-superscripts denote the coordinate frames: $\mathcal{L}$ for the LVLH frame and $\mathcal{I}$ for the inertial frame. Expanding Equation (2.9) leads to the Nonlinear Equations of

Relative Motion (NERMs)[5]:

$$\begin{aligned}
\ddot{x} - 2\dot{f}_c\dot{y} - \ddot{f}_c y - \dot{f}_c^2 x &= \frac{-\mu}{r_d^3}\big(r_c + x\big) + \frac{\mu}{r_c^2} + a_x \\
\ddot{y} + 2\dot{f}_c\dot{x} + \ddot{f}_c x - \dot{f}_c^2 y &= \frac{-\mu}{r_d^3} y + a_y \\
\ddot{z} &= \frac{-\mu}{r_d^3} z + a_z
\end{aligned} \tag{2.10}$$

where [2]:

$$\begin{aligned}
\ddot{f}_c &= \frac{-2\dot{r}_c\dot{f}_c}{r_c} \\
\dot{f}_c &= \frac{\|h_c\|}{r_c^2} \\
\ddot{r}_c &= r_c\dot{f}_c^2 - \frac{\mu}{r_c^2} \\
\dot{r}_c &= \dot{f}_c \frac{a_c e_c(1-e_c^2)\sin f_c}{\big(1 + e_c\cos f_c\big)^2}
\end{aligned} \tag{2.11}$$

The NERMs exactly describe the relative motion of a deputy satellite around a chief satellite. Since the two body problem was utilized, there is inherently an assumption that the Earth's gravitational force is the only external force acting on each of the two satellites. This implies the absence of disturbing forces such as those caused by solar radiation pressure, J_2 gravitational oblateness effects, aerodynamic drag forces, etc [2].

Ten states are needed to fully parameterize the NERMs, given in Table 5. Six of the ten parameters describe the position and velocity of the satellite with respect to the chief, and the remaining four describe the chief orbit. To solve the NERMs, typically numerical integration is required. No general closed-form solution exists, which motivated the need to develop linearized sets of dynamics to obtain closed-form solutions.

Table 5. Paramters to Describe the NERMs

Parameter	Description	Describes
x	Radial Position	Deputy w.r.t. Chief
y	In-track Position	Deputy w.r.t. Chief
z	Cross-track Position	Deputy w.r.t. Chief
$\dot{x}$	Radial Velocity	Deputy w.r.t. Chief
$\dot{y}$	In-track Velocity	Deputy w.r.t. Chief
$\dot{z}$	Cross-track Velocity	Deputy w.r.t. Chief
f	True Latitude	Chief w.r.t. ECI
$\dot{f}$	Time Rate of Change of True Latitude	Chief w.r.t. ECI
r_c	Radius of Chief's Orbit	Chief w.r.t. ECI
$\dot{r_c}$	Time Rate of Change of Chief's Orbit	Chief w.r.t. ECI

2.1.4 The Tschauner-Hempel Equations.

The NERM equations can be reduced down to the Tschauner-Hempel equations of motion by linearizing about a reference (chief) orbit by assuming [2]:

$$r_d \approx r_c \sqrt{\frac{2x}{r_c} + 1} \tag{2.12}$$

and converting the independent variable from time to true anomaly, f [6]:

$$\begin{bmatrix} \tilde{x} & \tilde{y} & \tilde{z} \end{bmatrix}^T = (1 + e\cos f) \begin{bmatrix} x & y & z \end{bmatrix}^T \tag{2.13}$$

After the independent variable transformation, denoted by the tilde, the Tschauner-Hempel equations of motion equations are [6]:

$$\begin{aligned} \tilde{x}'' - 2\tilde{y}' - \frac{3}{1 + e\cos(f)}\tilde{x} &= a_x \\ \tilde{y}'' + 2\tilde{x}' &= a_y \\ \tilde{z}'' + \tilde{z}' &= a_z \end{aligned} \tag{2.14}$$

where the prime operator denotes derivatives taken with respect to true anomaly, f. The Tschauner-Hempel equations of motion are useful for chief orbits of arbitrary

eccentricity. They can be further simplified into the Hill-Clohessy-Wiltshire (HCW) dynamics by assuming a circular chief reference orbit and converting the independent variable back to time [6].

2.1.5 The Hill-Clohessy-Wiltshire (HCW) Equations.

Hill-Clohessy-Wiltshire [7] show how the NERMs can be linearized about a circular chief orbit under the assumption of small separation distances between the deputy and chief relative to the chief's orbital radius and that the total time studied is sufficiently small compared to the total orbital period. As a rule of thumb, the total time should be less than the orbital period, given that error accumulates with both distance and time [8]. The result is a set of equations given by [7]:

$$\begin{aligned} \ddot{x} - 2n\dot{y} - 3n^2 x &= a_x \\ \ddot{y} + 2n\dot{x} &= a_y \\ \ddot{z} + n^2 z &= a_z \end{aligned} \tag{2.15}$$

where x, y, and z are the LVLH reference frame components of relative position. In Equation (2.15), n represents the mean motion of the chief orbit (comprised of the standard gravitational parameter μ, and the semi-major axis a). Note, the mean motion refers to the average angular frequency of a satellite about a closed orbit [9].

$$n = \sqrt{\frac{\mu}{a^3}} \tag{2.16}$$

The HCW equations consist of a set of three 2^{nd} order differential equations. The $\ddot{z}$ equation, being completely independent from the other two, implies that the cross-track motion is decoupled from the in-plane motion. Therefore, one can solve the in-plane motion separately from the out-of-plane motion. The solutions to the HCW equations are given by Alfriend et al. (with adapted notation)[2]:

$$
\begin{aligned}
x(t) &= \left[4x_0 + \frac{2}{n}\dot{y}_0\right] + \frac{\dot{x}_0}{n}\sin(nt) - \left[3x_0 + \frac{2}{n}\dot{y}_0\right]\cos(nt) \\
y(t) &= -\left[6nx_0 + 3\dot{y}_0\right]t + \left[y_0 - \frac{2\dot{x}_0}{n}\right] + \left[6x_0 + \frac{4}{n}\dot{y}_0\right]\sin(nt) + \frac{2}{n}\dot{x}_0\cos(nt) \\
z(t) &= \frac{\dot{z}_0}{n}\sin(nt) + z_0\cos(nt) \\
\dot{x}(t) &= \dot{x}_0\cos(nt) + \left[3x_0 n + 2\dot{y}_0\right]\sin(nt) \\
\dot{y}(t) &= -\left[6nx_0 + 3\dot{y}_0\right] + \left[6\dot{x}_0 n + 4\dot{y}_0\right]\cos(nt) - 2\dot{x}_0\sin(nt) \\
\dot{z}(t) &= \dot{z}_0\cos(nt) - z_0 n\sin(nt)
\end{aligned}
\tag{2.17}
$$

The HCW equations have formed the basis for modeling relative satellite motion that many authors have extended by adding in disturbances or eliminating various assumptions, as shown in Figure 4. For example by adding in the gravitational J_2 perturbation, Schweighart and Sedwick arrive at a form of the HCW equations that yields slightly better accuracy when there is a large J_2 effect, as shown by Sullivan [10]. Work done by Alfriend et al. [11] and De Bruin et al. [12] redefine the HCW equations with curvilinear coordinates, which show distinct improvements in modeling error if the satellites are separated along the direction of travel (i.e. along the $\hat{y}$ axis of the LVLH reference frame). Yamanaka and Ankersen [13] utilize the Tschauner-Hempel equations as a starting point for deriving a singularity free dynamical representation for elliptical relative satellite motion. By removing the circular orbit restriction, the Yamanaka-Ankersen State Transition Matrix (STM) is effective at modeling relative motion in Geosynchronous Transfer Orbits (GTO) and Molniya orbits, which the original HCW could not adequately model. And finally, by reparameterizing the orbit by relative orbital elements instead of position and velocity states, Gim and Alfriend utilize a technique to develop a dynamics model that accounts for both J_2 and eccentricity which yields at least an order of magnitude better accuracy than other types of formulations such as the HCW, and Yamanaka-Ankersen [10].

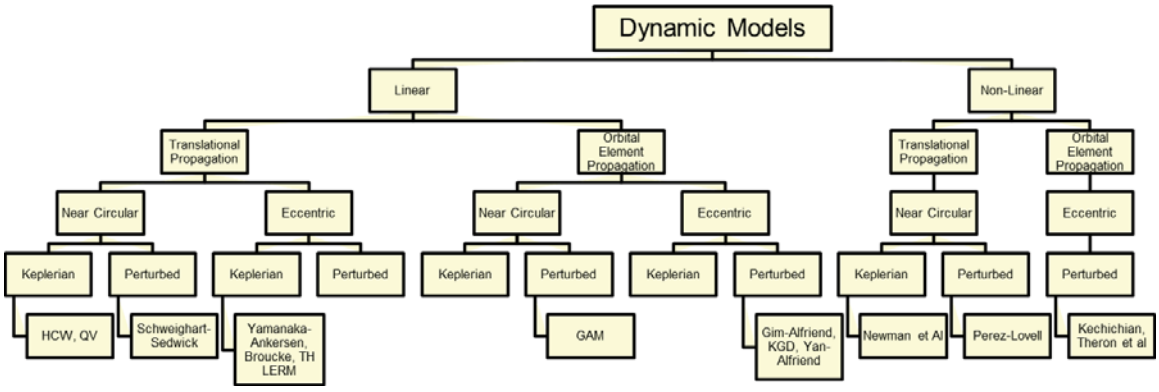


Figure 4. A Non-Exhaustive List of Dynamical Models for Relative Satellite Motion (adapted from [10])

2.1.6 Lovell's Relative Orbital Elements.

For certain velocities within the relative frame, there exists bounded Natural Motion Circumnavigation (NMC) trajectories that are periodic in nature and stable (excluding external disturbances). Shown in Figure 5, these NMC trajectories have a 2×1 elliptical shape, and are centered at the point (y_r, x_r). To satisfy a bounded relative NMC, the HCW equations require control in the in-track direction in order to control in-plane motion. Alfriend et al. [2] show that the condition to nullify drift in the in-plane motion is:

$$\dot{y}_0 = -2nx_0 \tag{2.18}$$

For circular chief orbits, this is known as the energy matching condition. Alfriend et al. [2] further states this energy matching condition establishes a local region of stability that is valid for small separation distances. Enforcing this condition creates bounded relative orbits in the LVLH frame that repeat indefinitely (neglecting real-world environmental perturbations). Thus these trajectories are extremely fuel efficient and are a common RPO technique in inspection missions, and formation keeping.

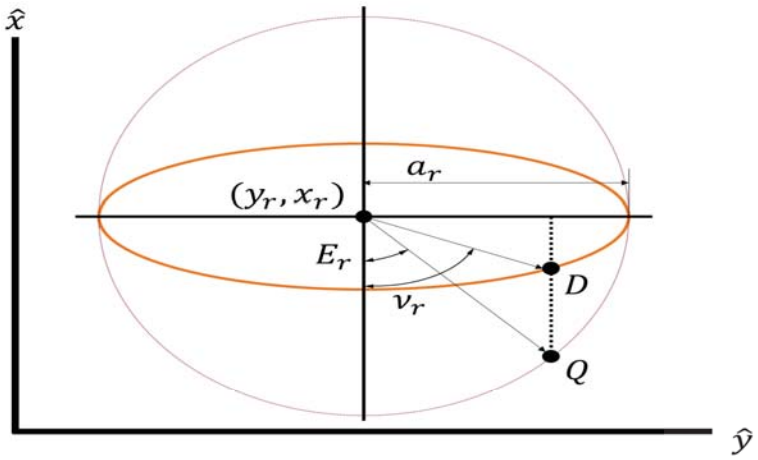


Figure 5. Lovell's ROEs, adapted from [14]

The work of both Lovell and Tragesser [15], and Lovell and Spencer [14] reparameterizes the HCW equations into parameters with geometric significance. These parameters provide intuitiveness that is invaluable in setting up relative motion orbits of a specific size and shape. They are given by [14]:

$$\begin{aligned}
x_r &= 4x_0 + \frac{2\dot{y}_0}{n} \\
y_r(t) &= y_0 - \frac{2\dot{x}_0}{n} - (6nx_0 + 3\dot{y}_0)(t - t_0) \\
a_r &= \sqrt{\left(6x_0 + \frac{4\dot{y}_0}{n}\right)^2 + \left(\frac{2\dot{x}_0}{n}\right)^2} \\
E_r(t) &= \text{atan2}\left(\frac{2\dot{x}_0}{n}, 6x_0 + \frac{4\dot{y}_0}{n}\right) + n(t - t_0) \\
A_z &= \sqrt{z_0^2 + \left(\frac{\dot{z}_0}{n}\right)^2} \\
\psi(t) &= \text{atan2}\left(z_0, \frac{\dot{z}_0}{n}\right) + n(t - t_0) \\
\nu_r(t) &= \tan^{-1}\left[2\tan\left(E_r(t)\right)\right] \\
\gamma(t) &= \psi(t) - E_r(t)
\end{aligned} \tag{2.19}$$

where the parameters are propagated in time [14]:

$$
\begin{aligned}
x_r &= x_{r,0} \\
y_r(t) &= y_{r,0} - \frac{3}{2} n x_{r,0}(t - t_0) \\
a_r &= a_{r,0} \\
E_r(t) &= E_{r,0} + n(t - t_0) \\
A_z &= A_{z,0} \\
\psi(t) &= \psi_0 + n(t - t_0)
\end{aligned}
\tag{2.20}
$$

In particular, these Relative Orbital Element (ROE) parameters also provide a convenient method for calculating the required initial velocity given either (y, x) position or the phasing angle, ν_r, shown in Figure 5.

Given a known semi-major axis of the NMC ellipse, a_r and the relative orbit phase angle, ν_r, Lovell's ROEs [14, 15] can be solved to get the required initial position (point D in Figure 5):

$$
\begin{aligned}
x_0 &= \frac{a_r}{\sqrt{\tan(\nu_r)^2 + 4}} \\
y_0 &= \frac{a_r \tan(v_r)}{\sqrt{\tan(v_r)^2 + 4}}
\end{aligned}
\tag{2.21}
$$

Once the radial and in-track components of position are known and assuming the center of the ellipse is coincident with the origin of the LVLH frame, the required velocity at point D is:

$$
\begin{aligned}
\dot{x}_0 &= \frac{n y_0}{2} \\
\dot{y}_0 &= -2 n x_0
\end{aligned}
\tag{2.22}
$$

2.2 Optimal Control

An optimal control problem deals with the determination of a function that minimizes or maximizes a given cost functional. Commonly, the solution of an optimal

control problem is applied as an open loop control law that, if followed, will optimize the objective (time, amount of control, etc). An optimal control problem is notionally structured as follows [16]:

$$\underset{\bar{u} \in \mathcal{U}}{\text{minimize}}\ J = \phi(t_0, t_f, \bar{x}(t_0), \bar{x}(t_f)) + \int_{t_0}^{t_f} \mathcal{L}\big(\tau, \bar{x}(\tau), \bar{u}(\tau)\big) d\tau \tag{2.23}$$

subject to the system dynamics

$$\dot{\bar{x}}(t) = \bar{f}(t, \bar{x}(t), \bar{u}(t))$$

subject to the path constraints

$$\bar{h}(t, \bar{x}(t), \bar{u}(t)) \leq \bar{0}$$

subject to the boundary constraints

$$\bar{g}(t_0, t_f, \bar{x}(t_0), \bar{x}(t_f)) \leq \bar{0}$$

where the states ($\bar{x}$), control ($\bar{u}$), and time (t) are bounded by

$$\bar{x}_{lower} \leq \bar{x}(t) \leq \bar{x}_{upper}$$

$$\bar{u}_{lower} \leq \bar{u}(t) \leq \bar{u}_{upper}$$

$$t_{lower} \leq t_0 < t_f \leq t_{upper}$$

and the initial conditions are bounded by

$$\bar{x}_{0,lower} \leq \bar{x}(t_0) \leq \bar{x}_{0,upper}$$

$$\bar{x}_{f,lower} \leq \bar{x}(t_f) \leq \bar{x}_{f,upper}$$

In these functions, $\mathcal{L}$ is the integrand of the running cost integral when written in the Bolza or Lagrange form. The Bolza form of the cost functional is a combination of the Mayer form (comprised of only terminal costs) and the Lagrange form (comprised of only cost integrals). The cost functional J can be easily converted between the three forms using the fundamental theorem of calculus [17]. The choice of $\mathcal{L}$ is

synonymous with picking a performance measure. There are a variety of forms of the cost functional J based upon the problem formulation. For example, in this research the minimum time formulation of the cost functional J is [6]:

$$J = t_f \tag{2.24}$$

In order to enforce the dynamics, i.e. $\bar{x} = \bar{f}\big(t, \bar{x}(t), \bar{u}(t)\big)$, the Hamiltonian $\mathcal{H}$ is introduced. Kirk [17] shows that the Hamiltonian is formulated as the sum of the running cost ($\mathcal{L}$) and the costates ($\bar{\lambda}$) multiplied by the dynamics, $\bar{f}(\bar{x}, \bar{u}, t)$:

$$\mathcal{H} = \mathcal{L} + \bar{\lambda}^T \bar{f}(\bar{x}, \bar{u}, t) \tag{2.25}$$

The costates in Equation (2.25) are continuous analogues to the adjoint Lagrange multipliers and are added to the cost functional J according to:

$$J = \int_{t_0}^{t_f} \left\{ \mathcal{L}(\tau, \bar{x}(\tau), \bar{u}(\tau)) + \left[\frac{\partial \phi(\bar{x}, \tau)}{\partial \bar{x}}\right]^T \dot{\bar{x}} + \frac{\partial \phi(\bar{x}, \tau)}{\partial \tau} + \bar{\lambda}^T [\bar{f}(\bar{x}, \bar{u}, \tau) - \dot{\bar{x}}] \right\} d\tau \tag{2.26}$$

The three necessary conditions for optimal control are then derived from Equation (2.26) through the use of the Calculus of Variations. Whenever these conditions hold true for admissible, bounded control, then the candidate function is an extrema [18]:

$$\begin{aligned}
\dot{\bar{x}}(t) &= \frac{\partial \mathcal{H}}{\partial \bar{\lambda}}(\bar{x}^*(t), \bar{u}^*(t), \bar{\lambda}^*(t), t) \\
\dot{\bar{\lambda}} &= -\frac{\partial \mathcal{H}}{\partial \bar{x}}(\bar{x}^*(t), \bar{u}^*(t), \bar{\lambda}^*(t), t) \\
\mathcal{H}(\bar{x}^*(t), \bar{u}^*(t), \bar{\lambda}^*(t), t) &\leq \mathcal{H}(\bar{x}^*(t), \bar{u}(t), \bar{\lambda}^*(t), t) \\
0 &= \left[\frac{\partial \phi}{\partial \bar{x}}\big(\bar{x}^*(t_f), t_f\big) - \bar{\lambda}^*(t_f)\right]^T \delta \bar{x}_f \\
&\quad + \left[\mathcal{H}(\bar{x}^*(t_f), \bar{u}^*(t_f), \bar{\lambda}^*(t_f), t_f) + \frac{\partial \phi}{\partial t}(\bar{x}^*(t_f), t_f)\right] \delta t_f
\end{aligned} \tag{2.27}$$

The above conditions are necessary for bounded optimal control policies; however, they are not sufficient conditions. The sufficient conditions for optimal control are needed in order to determine if an extrema constitutes a local minimum, maximum, or saddle point solution. These sufficient conditions state that the first partial derivative of $\mathcal{H}$ with respect to the control is equal to zero and the second derivative matrix, the Hessian, is positive definite [17]:

$$\begin{aligned} \frac{\partial \mathcal{H}}{\partial \bar{u}}(\bar{x}^*(t), \bar{u}^*(t), \bar{\lambda}^*(t), t) &= 0 \\ \left[\frac{\partial^2 \mathcal{H}}{\partial \bar{u}^2}\right] &> 0 \end{aligned} \tag{2.28}$$

Thus, the objective of the optimal control problem is to find the admissible control, $u^*(t)$, that satisfies Pontryagin's Minimum Principle for bounded control [19]:

$$\bar{u}^*(t) = \underset{\bar{u}^* \in \mathcal{U}}{\operatorname{argmin}} \ \mathcal{H} \tag{2.29}$$

Thus, the optimal control $\bar{u}^*(t)$, within the admissible set $\mathcal{U}$, minimizes $\mathcal{H}$. In contrast, if we wished to maximize an objective, one would simply minimize the negative of $\mathcal{H}$ [17, 18].

2.2.1 Solving Optimal Control Problems.

Some optimal control problems can be solved analytically, however most real-world problems are often too complicated, or would take too much time to determine a solution. To aid the solution finding process, a variety of different algorithms exist, which include direct, indirect, and metaheurstic methods [16]. An indirect method analytically derives the necessary conditions to ensure mathematical optimality, namely those in Equation (2.27). If solvable, an indirect method produces a guaranteed local optimal solution, however the difficulty of deriving the necessary conditions varies on

the complexity of the problem [16]. For example, if the derivative information cannot be analytically solved for with a closed-form solution, perhaps because of discontinuities, or singularities, then the derivation of the analytic conditions for optimality may prove infeasible unless a numerical approximation for the derivatives can be found [3].

Indirect methods usually involve solving a two-point boundary value problem [3], which is defined as a differential equation(s) that contain a set of constraints. Conway [3] notes that these constraints are usually a combination of boundary conditions on the states and costates at the initial time and final time. For example, Lambert's problem is a well known two-point boundary value problem where an initial and terminal state is given, and a trajectory is sought that passes through both points assuming two-body dynamics [20]. A closed-form solution to Lambert's problem is not possible, however it is easily solved with indirect numerical methods [20]. Furthermore, given that costates are often non-physical, and can have values that are orders of magnitude different than the states, one cannot intuitively select initial costates for the problem [3]. Thus, indirect methods can provide a way to numerically solve an optimal control problem, but are sometimes too cumbersome because they require values at the initial and terminal states which may not be fully known, or else are being optimized [3].

Direct methods eliminate the need for finding analytical conditions for optimality by approximating the optimal control solution and then continually refining it until the solution converges within the error specification defined by the user. For example, Conway [3] shows how one direct optimization method converts the continuous optimal control problem into parametric representations of the control and state histories (either discretely or continuously through polynomial approximation). A collocation method is used to transcribe the optimal control problem into a nonlinear program

(NLP) by creating a series of points on a mesh where the governing equations are satisfied at each collocation point [3]. States and controls in between the collocation points are approximated with various interpolation techniques. The candidate solution is then obtained through NLP solvers such as the Interior Point Optimizer (IPOPT) [21] and the Sparse Nonlinear Optimizer (SNOPT) [22].

Pseudospectral methods are a type of direct optimization numerical method. Conway [3] provides an example of how one type of pseudospectral method employs Lagrange interpolating polynomials to create a polynomial expression for the state history (in contrast to the Hermite-Simpson rules typically used in indirect methods). These Lagrange interpolating polynomials are utilized to calculate the error in approximation through the use of differentiation matrices. The resulting error from the derivative computation is then reduced by either picking more interpolation points for the Lagrange interpolation polynomial or by increasing the order of the polynomial until the error drops below the required user specification [3]. One downside to pseudospectral solvers is that, like many direct methods, an initial guess is often required to start the approximation process [23].

The last category of optimization methods discussed here are called "metaheuristic", which are often inspired by biology. Metaheuristical optimization techniques employ optimal search methods to find a local extrema for a given cost function. For example, the particle swarm algorithm is a metaheuristic method that was originally developed by Kennedy and Eberhart [24], and inspired by flocks of birds searching for food. Kennedy and Eberhart [24] formulate a collection of particles with three properties which influence how the particle moves around the solution space. These properties can be thought of as components of a velocity vector that help shape the direction the particle will move in one iteration. The first property is coined the "inertial" component and serves to keep the particle moving in the "same direction"

as the last iteration. This inertial component of velocity is counter-acted by the other two properties: cognitive and social. The cognitive component of the velocity vector points to the location of the particles best fitness (the minimum value of the cost functional that the individual particle has come across). The social component of the velocity vector is similar to the cognitive component, except that it points in the direction of the group's best fitness. These velocity components for the i^{th} particle at the j^{th} iteration are expressed as [19]:

$$\begin{aligned} V_k^{j+1}(i) =& c_1 \bar{V}_k^j(i) + c_2 R(0,1)\left[\bar{\Psi}_k^j(i) - \bar{r}_k^j(i)\right] \\ &+ c_3 R(0,1)\left[\bar{\rho}_k^j(i) - \bar{r}_k^j(i)\right] \end{aligned} \tag{2.30}$$

where $\bar{\Psi}_k^j(i)$ contains the individual particles best fitness location, $\bar{\rho}_k^j(i)$ contains the swarm's best fitness location, $R(0,1)$ are randomized variables ranging from 0 to 1, and c_1 through c_3 are constant scalar tuning terms. A particle swarm optimization's purpose is to find a set of parameters, or coefficients in the given cost function, that will produce a minimum cost function result. As the particle moves around the solution space the cost function is continually evaluated at each j^{th} iteration, which provides a measure of fitness that is compared to the individual particle's and group's best result. These search parameters are often final time and other parameters from necessary conditions for optimally [25].

2.3 Zero-Sum Differential Games

Differential game theory is the mathematical formulation of the possible outcomes resulting from the interaction of two or more players having different (conflicting) objectives. These players are assumed to be intelligent in that they will utilize optimal strategies and have perfect information about the other player. Making these assumptions narrows the scope from all possible outcomes to the optimal outcomes

that represent the best action for each player. Isaacs [26] describes the objective of the game as the pay-off, which is a measurable value of the particular end state of the differential game for each player. In this research, consistent with much of literature, the pay-off will be the value of the cost functional.

Differential game theory applied to space yields a plethora of literature, which is summarized in Table 6. At the heart of every article is an optimal control problem. The differential games studied exist in two large categories: deterministic and non-deterministic (stochastic). A deterministic game considers scenarios where there is perfectly complete information shared between the two players. In other words, each player knows the other's state (such as position and velocity) at each instance of time. In contrast, the main difference in stochastic games is the incorporation of uncertainty, which can be implemented in a variety of places such as in the state measurements, the assumed orbital dynamics, and the other player's control law coefficients. Stochastic games are almost always paired with an estimation algorithm that takes in measurements and produces a single estimated value of the true measurement with a new confidence bound. The end result of analyzing stochastic games is to add realism by incorporating things such as measurement uncertainty and process noise into the solution process that deterministic games lack.

The solutions to pursuit-evasion games have been found using a variety of different techniques, which can be broken down into three large categories: direct, indirect, and metaheuristical optimization methods as discussed in Section 2.2.1. Using an indirect heursitc method, Pontani and Conway [25] solve the optimization problem by satisfying the relative magnitude of the costates and showing how the transversality condition is ignorable in certain circumstances. Stupik [19] utilizes this indirect heuristic method along with a particle swarm optimization routine to solve a pursuit-evasion differential game. Stupik utilizes simplified dynamics, which is taken

by Prince et al. [27] and applied to elliptical orbits with more complex dynamics. Taking this a step further, Shen et al. [28] formulate a pursuit-evasion differential game using a high fidelity SGP4 propagator where the pursuer and evader start in non-coplanar orbits. Shen et al. show how a pursuer spacecraft can become coplanar and then catch the evader in a zero-sum game within a high fidelity NASA General Mission Analysis Tool (GMAT) simulator tool [28]. While not directly stated by Shen et al., this work validates the use of linearized dynamics for pursuit-evasion games because Shen et al. show that similar solutions can be gained from both the linearized and nonlinear set of dynamics. Taking a different approach to solving the 3D pursuit-evasion game, Pontani and Conway [29] utilize a semi-direct collocation with nonlinear programming (semi-DCNLP) to solve a two-sided path optimization problem. This solution technique was originally developed by Horie and Conway [30], which uses indirect optimization methods to derive optimality criteria for one player to include in the other's NLP solution process. A genetic algorithm provides an initial guess at the solution, which is iterated upon by the NLP until it converges to a final solution [30]. Liu et al. [31] takes this semi-DCNLP method further by expanding it to a three player differential game of the lady, bandit, and body-guard [32], in which the lady is a non-maneuvering spacecraft that the bandit spacecraft intends to intercept while the body-guard spacecraft intends to protect. Liu et al. [31] show how the semi-DCNLP architecture can handle multiple objectives of the two players; that is they must interact with one another while also attempting to intercept/protect the lady spacecraft.

In these deterministic games, a zero-sum differential game condition occurs when both players employ optimal strategies towards a common, but opposing goal [33]. Jagat [5] shows that these zero-sum differential games satisfy the Nash equilibrium, which is inherently a saddle-point solution. Therefore, it follows that each player can

only employ the optimal strategy if there is perfect and shared information between the two players, since each player needs to know what the other player is doing at all times in order to maintain the best strategy. Carr [33] reiterates this point by showing how the Nash equilibrium results in the best possible outcome for the evader, and moving off the saddle-point solution through the use of a sub-optimal strategies results in a quicker time to capture. In this way, Carr [33] and Prince [27] show that the results of the Nash equilibrium can be viewed as a performance ceiling from which one can judge various sub-optimal strategies. For example, if the evader performed any action other than the optimal strategy, they would be captured more quickly than the optimal control solution. With the incorporation of uncertainty into the shared states between the two players, sub-optimal strategies could arise. These sub-optimal results can be compared back to their analogous perfect information game in order to determine how sub-optimal the outcome became as a result of adding in uncertainty.

Stochastics is introduced into deterministic pursuit-evasion games to add realism by accounting for sources of inaccuracy. Stochastic pursuit-evasion games are typically subdivided into three smaller categories: incomplete information, imperfect information, and uncertain information [34]. Incomplete information games deal with uncertainty in the control strategy employed by the other player, which is often then estimated by a type of filter. Imperfect information games deal with uncertainty in the shared states between the players, whereas uncertain information games deal with uncertainty in the system dynamics [34]. By examining an incomplete, imperfect information differential pursuit-evasion game Woodbury and Hurtado [34] utilize an Unscented Kalman Filter (UKF) to estimate both the strategy and the state of the other player. Woodbury and Hurtado [34] form the basis for their control estimation process off the work done by Satak [35], who shows how estimation theory can be utilized in a behavior learning sense to predict the other player's control inputs if an

assumed form of control (such as the Riccati equation) is utilized. Woodbury and Hurtado [34] then apply a UKF to refine the full state measurements taken with white Guassian noise. Cavalieri et al. [36] also utilize Satak's behavior learning algorithm [35] to estimate the impact of uncertain dynamics on the pursuer's trajectory. Maloy and Lee [37] study a pursuit evasion game in the polar coordinate frame where each player makes noisy (imperfect) measurements on the other using sensors capable of measuring range, range rate, and bearing angles. Amato et al. [38] formulate a game based off the solution of two scaled Riccati differential equations where each player must model the dynamics of the system, implying a level of uncertain information to the scenario that's different from each player's perspective. Willman [39] derives a class of pursuit evasion scenarios affected by noisy (imperfect) measurements that under certain assumptions can be related back to their analogous deterministic scenario without stochasticity. Behn and Ho [40] show how a pursuer with perfect information is able to close in on an evader with noisy (imperfect) information, demonstrating the power of having access to less noisy measurements. Similarly, Rhodes and Luenberger [41] add noise to one or both player state measurements, and formulate optimal closed-loop controllers for each case. Finally, Antoniades et al. [42] show how a team of pursuers with only a limited view of the total simulation space (imperfect information) can utilize probability mapping theory to minimize the overlap of their search patterns for the evaders.

2.3.1 Knowledge Gap.

Of all the work pertaining to stochastic space based pursuit-evasion differential games surveyed, the focus has been on employing filtering algorithms to produce optimal control solutions. This research approaches the imperfect stochastic pursuit-evasion game differently by focusing not on a particular algorithm for solving stochas-

Table 6. Survey of PE Games

Researcher	Incomplete	Imperfect	Uncertain	Perfect
Stupik [19]				×
Prince et al. [27]				×
Shen et al. [28]				×
Pontani and Conway [29]				×
Horie and Conway [30]				×
Liu et al. [31]				×
Rusnak [32]				×
Carr et al. [33]				×
Jagat [5]	×		×	
Woodbury and Hurtado [34]	×	×		
Satak [35]	×			
Cavalieri et al. [36]	×		×	
Maloy and Lee [37]		×		
Amato et al. [38]			×	
Willman [39]		×		
Behn and Ho [40]		×		
Rhodes and Luenberger [41]		×		
Antoniades et al. [42]		×		

tic games, but instead on analyzing the entire stochastic solution set with statistical regression techniques. A collection of inputs and outputs from a large set of optimal control solutions will be used, along with robust multivariate regression techniques, to produce a linear regression model that maps the initial pursuer position to final capture position. With this model, a stochastic scenario will be analyzed where an evader is uncertain of the true pursuer's starting position. In this scenario, it is assumed that the pursuer is on an NMC trajectory about the evader, and the pursuer's starting position is inside a given covariance ellipsoid known to the evader. A parameter study of this scenario is performed by modifying elements of the scenario, such as the size of the initial covariance ellipse, and implementation of an evader performance scale factor (which handicaps the evader's available control acceleration). The results of the parameter study are geared towards finding advantageous/disadvantageous matchups to increase mission planning effectiveness for future space operations. The linear regression model by itself is also useful for forecasting pursuit-evasion interceptions

for mission planning purposes when only the terminal position and time are required (since the control profile is not calculated). Thus, this research begins to bridge the gap between deterministic and stochastic pursuit-evasion differential games by utilizing deterministic games to analyze stochastic effects [43].

2.4 Closed-Loop Control Theory

Thus far, everything discussed has been open loop in nature. Closed-loop designs are needed to apply estimation theory, as well as applications of controllers to drive the error associated with the difference in a reference trajectory and the actual trajectory to zero.

2.4.1 The Linear Quadratic Regulator (LQR) Controller.

A typical linear, time-invariant, state-space dynamical system is given by [44]:

$$\dot{\bar{x}} = [A]\bar{x} + [B]\bar{u} \tag{2.31}$$

where $\bar{x} \in \mathcal{R}^n$ and $\bar{u} \in \mathbb{R}^m$. A simple closed-loop control scheme can be chosen such that $\bar{u} = -[K]\bar{x}$, where $[K]$ is a carefully selected constant scalar matrix whose size depends upon the size of the control vector and number of states. In general, $[K]$ is size $m \times n$. The dynamical system then becomes:

$$\begin{aligned} \dot{\bar{x}} &= [A]\bar{x} + [B]\big(- [K]\bar{x}\big) \\ &= \big([A] - [B][K]\big)\bar{x} \end{aligned} \tag{2.32}$$

whose characteristic equation (in the s domain) is [44]:

$$\det\Big[s[I] - [A] + [B][K]\Big] = 0 \tag{2.33}$$

The control system is then stable so long as the roots of the characteristic equation are negative, and non-repeating on the imaginary axis of the $j\omega$ plane. The optimal value of $[K]$, is found with the given cost functional, J, given by [44]:

$$J = \frac{1}{2}\int_0^\infty (\bar{x}^T[Q]\bar{x} + \bar{u}^T[R]\bar{u})dt \tag{2.34}$$

In this formulation, the term $\bar{x}^T[Q]\bar{x}$ measures the cost associated with error, whereas the term $\bar{u}^T[R]\bar{u}$ measures the cost associated with amount of control used. The matrices $[Q]$ and $[R]$ are then user chosen constant weighting matrices that allow the user to prioritize the importance in reducing the error compared to the amount of control utilized in the final solution's value for $[K]$. Note that since there are assumed 6 states $(x, y, z, \dot{x}, \dot{y}, \dot{z})$ the performance matrix $[Q]$ is a 6×6 square matrix. Similarly with an assumed control in x, y, z, direction, the control matrix $[R]$ is a 3×3 square matrix. The user may specify any value for the $[Q]$ and $[R]$ matrices, so long as they are symmetric, $[Q]$ is positive semi-definite, and $[R]$ is positive definite [17, 44]. Thus, the algebraic Riccati equation becomes [45]:

$$[P][A] + [A]^T[P] - [P][B][R]^{-1}[B]^T[P] + [Q] = 0 \tag{2.35}$$

where $[K] = [R]^{-1}[B]^T[P]\bar{x}$. An appropriate choice of the $[K]$ matrix is thus based upon the solution to the algebraic Riccati equation and the constant covariance matrix $[P]$ [44]. The presented LQR control method will be chosen in the GNC architecture because of its ease in implementation, and that its linearization assumptions already exist because of the use of HCW and TH dynamics. Another advantage of the LQR, over heuristically selecting a value for the gain $[K]$ is that the LQR methodology guarantees both stability and optimal performance [17].

2.4.2 Estimation Theory - Filters.

The primary purpose of any filter is to reduce a particular aspect in the input. In the context of control theory, it is desired to reduce measurement and/or process noise. In this research, additive, white Gaussian noise will be added to the states in order to simulate imperfections in the way data is gained from sensors, limitations on accuracy from hardware, etc. Maybeck describes the concept of white noise as time invariant and infinite in power [46]. In other words, Maybeck explains that white noise is assumed constant power across all frequencies (hence infinite in power), and knowledge of the value of measurement noise at one instance in time does not allow you to predict the value of measurement noise at any other time (wide sense stationary processes)[46].

2.4.3 Filters vs. Observers.

Maybeck [46] describes a filter as an algorithm that processes data given as an input. This data is usually in the form of past measurements, which the filter processes in order to estimate future measurements, or to estimate variables that cannot be directly measured by a sensor. Similarly Ogata describes a full-order observer, shown in Figure 6, as a type of estimator that utilizes the same dynamics as the plant to propagate and estimate the missing unmeasured state(s). An observer is deterministic in nature, and by definition it does not include schochastics. This constitutes a fundamental difference between an observer and a filter; an observer is inherently deterministic, and a stochastic observer is another name for a filter [47].

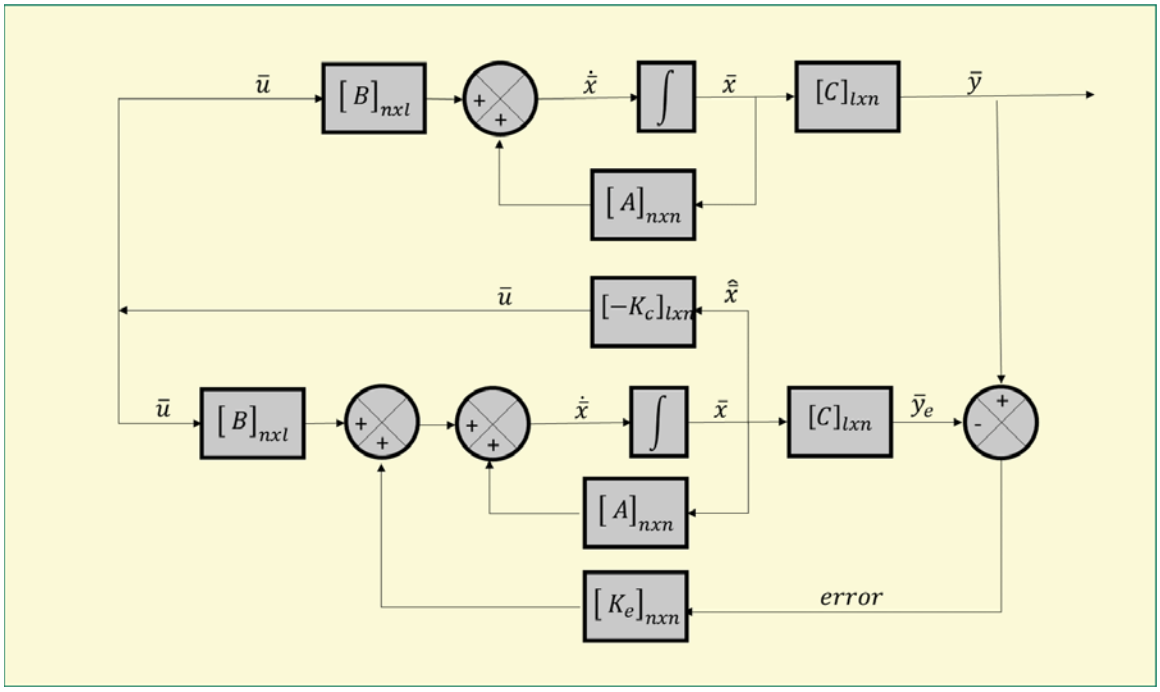


Figure 6. Full-State Observer [47]

2.4.4 The Linear Kalman Filer.

The Linear Kalman Filter (LKF) is defined by Maybeck as an "optimal recursive data processing algorithm" [46]. In other words, the LKF is a filter that optimally drives the mean squared error to zero. Maybeck asserts that one aspect of optimality that an LKF provides is that all information that is input into the filter is utilized, which is optimal in the sense that no data is wasted. Maybeck notes another strength of Kalman filter designs is that they do not require the entire data history, but instead only the last measurement, which is propagated to the new time step when the new measurement is taken. In other words, as time progresses the past measurement becomes less accurate in predicting the current state. This idea is illustrated by Figure 7, which shows how the error in the last estimate's probability density function steadily increases over time, lowering its overall usefulness as its amplitude drops and the base stretches outward to cover larger possible values of x. Note, in Figure

7 the data spread is notional, and should not always to be expected to be uniform, symmetric, etc. It is a common assumption, however, to assume a normal distribution of data when the true distribution is not known [48].

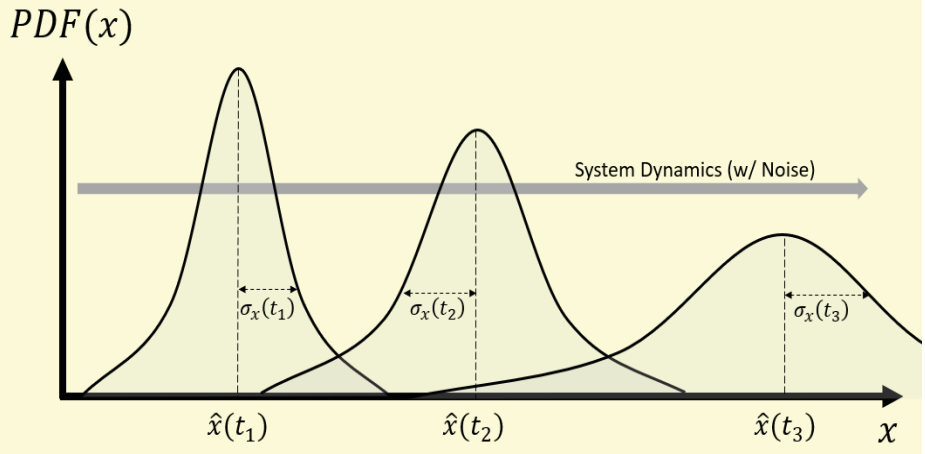


Figure 7. Propagation of Estimate Probabilities through System Dynamics, adapted from [46]

The linear Kalman filter begins by initializing the Kalman gain with a set of user defined initial estimates. These estimates consist of the state covariance $[P]$, known at time k and the measurement covariance, $[R]$. The Kalman gain is given by [49]:

$$[K]_k = [P]_k^{'}[H]^T\left([H][P]_k^{'}[H]^T + [R]\right)^{-1} \tag{2.36}$$

where the prime operator denotes previous iteration information. When the filter is started, the initial covariance estimate take the place of the previous measurement in Equation (2.36). When a set of new measurements are given as inputs to the LKF, the estimation is updated [49]:

$$\hat{x}_k = \hat{x}_k^{'} + [K]_k\left(z_k - [H][x]_k^{'}\right) \tag{2.37}$$

where the term $\left(z_k - [H][x]_k^{'}\right)$ is called the "innovation" or "measurement residual". Next, the covariance $[P]_k$ for the k^{th} iteration is calculated [49]:

$$[P]_k = \left([I] - [K]_k[H]\right)[P]_k^{'} \tag{2.38}$$

With the current iteration's covariance estimate, the $k+1$ estimate can be found from the k^{th} estimate[49]:

$$\begin{aligned} \hat{x}_{k+1} =& [\Phi]\hat{x}_k \\ [P]_{k+1} =& [\Phi][P]_k[\Phi]^T + [Q] \end{aligned} \tag{2.39}$$

where Φ is the state transition matrix (obtainable by solving the matrix exponential for linear time invariant systems) and $[Q]$ is the process noise matrix. A new Kalman gain can then be calculated with the newly projected $k+1$ state and covariance estimates.

2.4.5 The Extended Kalman Filter.

The inherent linearity assumption in a LKF limits its applicability to nonlinear estimation. It is best practice to assume that a linearization is only accurate about the linearization point for short periods of time and thus not well suited for estimation of nonlinear systems. An Extended Kalman Filter (EKF) takes the same process framework of a LKF, but linearizes the state and measurement functions about the point of estimation. By utilizing the Jacobian matrix, a first order approximation of the nonlinear functions can be obtained, which increases the accuracy of the estimation process. Thus, the EKF propagates the mean and covariance through linearized state and measurement functions rather than assuming a set of linear functions as in the LKF [50].

2.4.6 The Unscented Kalman Filter.

The Unscented Kalman Filter (UKF) was developed to handle nonlinear measurement models through the use of the unscented transform, and thus avoiding the measurement linearization process performed within the EKF [50]. Julier and Uhlmann show how their unscented transform method approximates the probability distribution rather than approximating the transformation function [51], which by doing so avoids taking the Jacobian derivatives normally done within an EKF. The unscented transform method utilizes an optimal $2n+1$ number of sigma points to propagate the first two statistical moments (the mean and covariance) through the full nonlinear measurement transformation, which has a distinct number of advantages. First, the calculation of the Jacobian for a nonlinear system could be intractable or subject to human error when coding, transcribing, and/or solving. Second, the unscented transform can be both rotated and weighted to account for things such as sensor bias, and higher order information like the third statistical moment (skewness). Ultimately the UKF is demonstrated to be on the same order of computational speed as the EKF, while providing more accurate measurement estimates compared to simulated truth data for certain cases [51].

2.5 Regression and Interpolation Techniques

2.5.1 Regression Techniques.

When an analytical function describing the relationship between two quantities is not available, empirical equations from identified data trends are often the next best option. One method of obtaining empirical equations is through statistical regression. Regression is another name for curve fitting, in which a function is found to identify a data trend in collection of points. The goal of regression is to determine the coefficients

of β that minimize the distance between the function and each point. Linear regression assumes that the β coefficients are linear; that is, the β terms are not raised to a power, nor multiplied by other β's. It is important to note that linear regression does not mean that the regression produces only straight lines. For example, linear regression can be expressed as [52]:

$$f(x, y) = \beta_0 x^2 + \beta_1 x + \beta_2 \tag{2.40}$$

Equation (2.40) is polynomial in nature, however the β coefficients vary linearly. There are a variety of methods to determine the values for β, such as ordinary least squares, weighted least squares, and generalized least squares [52]. In all these methods, it is sought to minimize the summation of the squared residuals. A residual in this context is simply the vertical distance between the point and the fitting function. The coefficient of determination, r^2, is a widely used metric of measuring the variance in the residuals, which is given by the following formula [53]:

$$r^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{2.41}$$

where $\bar{y}$ is the mean and $\hat{y}$ are the calculated values of y from the fitting function. The coefficient of determination ranges from zero to one. An r^2 value of 1.0 corresponds to no variance in the calculated values of y, meaning that the regression results exactly fit the data points.

Matlab will be used to compute the linear regressions given a collection of points from the optimal control solutions. These linear regression models will form the underlying structure of the models built for Section 3.1 and 3.2. With these linear regression models, an input to output mapping can be made to replace the optimal control solution process, which is described in more detail in Section 3.1.3.

2.5.2 Interpolation Techniques.

In addition to regression techniques, interpolation methods will also be used to augment the linear regression models. After a linear regression model has been built, it can be evaluated like an ordinary function. In this way the linear regression model can estimate results for arbitrary points around its creation data, thereby providing results in between the data points used to construct it. However if a linear regression model was built for a specific evader performance scale factor, the model cannot predict values for any other evader performance scale factor. If the requested evader performance scale factor falls in between the values of two linear regression models, then the results of both those models can be interpolated between to provide results for the requested evader performance scale factor. In its simplest form, a straight line is drawn between the two values: (x_1, y_1) and (x_2, y_2) the requested value is found for y, given x:

$$y = \frac{y_1(x - x_2) - y_2(x - x_1)}{x_1 - x_2} \tag{2.42}$$

The accuracy of this result depends upon the separation distance between the two points used in the interpolation process, and can be improved further still by including more points. Davis [54] states that a straight line passing through 2 points can be interpolated for an interior point. He goes on to state that a more accurate result can often be obtained by passing a parabola through 3 points, and similarly a cubic polynomial through four points, [54]. Using this information, and taking the idea of straight line interpolation to three dimensions results in bilinear interpolation. Whereas before the function, f, depending upon only one independent parameter, x, consider the case where it depends upon two independent parameters: $z = f(x, y)$. Depicted in Figure 8, if there are known z values at points (x_1, y_1), (x_1, y_2), (x_2, y_1), and (x_2, y_2), then multiple straight interpolations can be performed to obtain estimates for arbitrary requested values (x, y). Thus, as stated by Davis [54], four points

will be required to find the interior point because straight line interpolation requires two points each, and there are two lines.

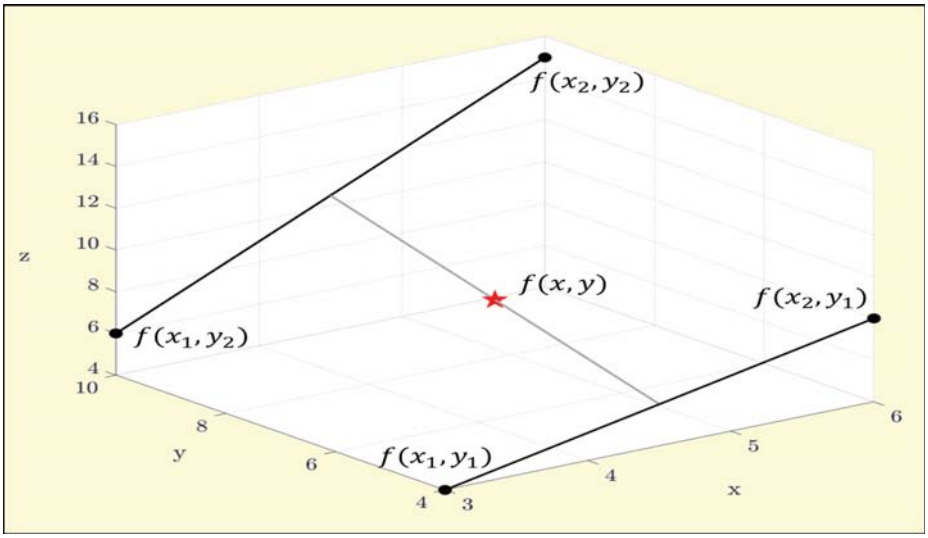


Figure 8. Bilinear Interpolation

Following Wang [55] (with adapted notation), interpolating in the first direction yields:

$$\begin{aligned} f(x, y_1) &= \frac{x - x_1}{x_2 - x_1}\Big(f(x_2, y_1) - f(x_1, y_1)\Big) + f(x_1, y_1) \\ f(x, y_2) &= \frac{x - x_1}{x_2 - x_1}\Big(f(x_2, y_2) - f(x_1, y_2)\Big) + f(x_1, y_2) \end{aligned} \tag{2.43}$$

then in the second direction [55]:

$$f(x, y) = \frac{y - y_1}{y_2 - y_1}\big[f(x, y_2) - f(x, y_1)\big] + f(x, y_1) \tag{2.44}$$

Expanding upon this concept, Chang [56] shows that bilinear interpolation is a form of surface estimation, called trend surface analysis. Through the use of statistics, Chang [56] shows how bilinear interpolation approximates the surface given by:

$$f(x, y) = b_0 + b_1 x + b_2 y \tag{2.45}$$

Through the use of statistics for n arbitrarily arranged points, one can form the matrix [56]:

$$\begin{bmatrix} n & \sum_{i=1}^{n}(x) & \sum_{i=1}^{n}(y) \\ \sum_{i=1}^{n}(x) & \sum_{i=1}^{n}(x^2) & \sum_{i=1}^{n}(xy) \\ \sum_{i=1}^{n}(y) & \sum_{i=1}^{n}(xy)\sum_{i=1}^{n}(y) & \sum_{i=1}^{n}(y^2) \end{bmatrix} \begin{bmatrix} b_0 \\ b1 \\ b2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n}(z) \\ \sum_{i=1}^{n}(xz) \\ \sum_{i=1}^{n}(yz) \end{bmatrix} \tag{2.46}$$

Equation (2.46) can then be solved for the b coefficients and plugged into Equation 2.45 to obtain an estimate for the interpolation. This formulation is an improvement over the formulation shown by Wang [55] since more than four points can be used to increase the accuracy of the result. For example, Figure 9 depicts a surface with known points arranged in a grid pattern. The intersection of the line and the surface indicates a point at which the two presented bilinear interpolation methods will estimate using the surrounding data. Table 7 shows the comparison in accuracy for the two methods.

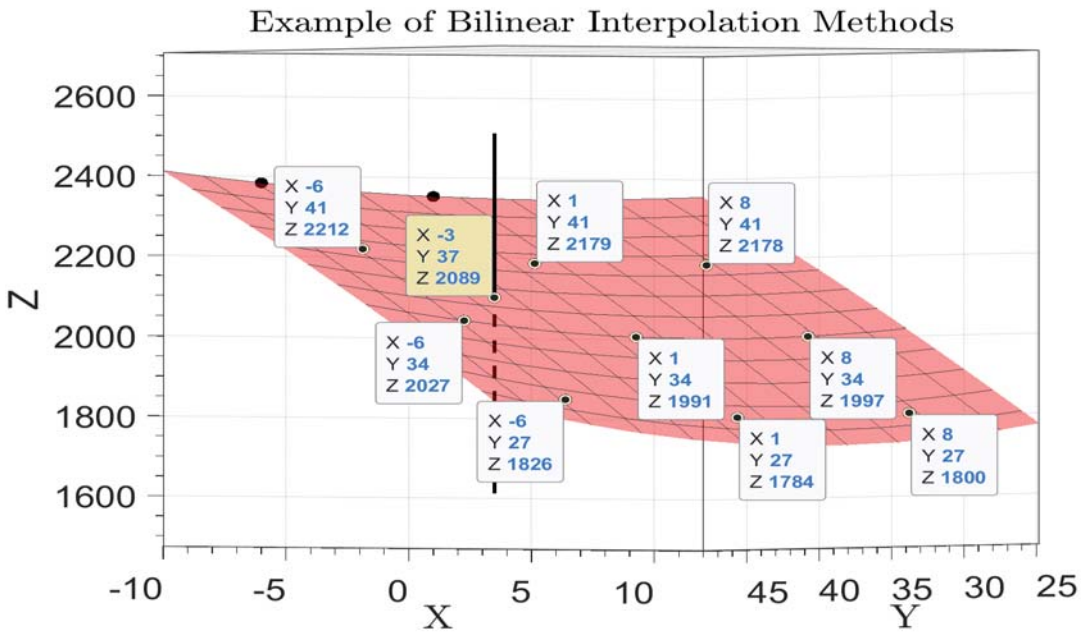


Figure 9. Bilinear Interpolation Example

Table 7. Effect of Using More Sample Points in Interpolation

Method	z Value	% Diff
Wang's Method:	2091.408	0.123%
Chang's Method (4 pts):	2091.393	0.122%
Chang's Method (5 pts):	2089.571	0.035%
Chang's Method (6 pts):	2089.488	0.0311%
Chang's Method (7 pts):	2089.294	0.0217%
Truth:	2088.839	

Bilinear interpolation will be used to interpolate between linear regression models that have two independent parameters. Similar to linear interpolation, bilinear interpolation's accuracy is dependent upon the distance between the chosen interpolated points. Therefore, for the purposes of this research interpolation techniques will only be employed when needed and always with nearest neighbor points.

2.6 Chapter Summary

A background in all areas necessary for understanding relative satellite motion, as it applies to pursuit-evasion differential games, has been given. Through extensive literature review, many solution algorithms have been shown to exist. This research will seek to create a solution algorithm ideal for on-board computation by utilizing regression techniques on a generated data set. The use of regression techniques solves the computational challenges associated with most existing algorithms by providing a computationally efficient method of calculating a new solution based upon known data trends. The next section will examine in detail how to build a linear regression model for a notional pursuit-evasion differential game based upon validated data, and then showcase its use in a large scale parameter study.

III. Methodology

This chapter describes the methodology used to formulate the four specific problems covered in more detail in the separate sections below. First, a methodology for solving a two-player differential game is introduced. This work will utilize the algorithm developed by Prince et al. [27]. For Problem A, the pursuit-evasion differential game solution algorithm will be modified to work as a function, and then run iteratively to create layers for a 2D linear regression model. This linear regression model will be crafted to support arbitrary pursuer positions on a bounded relative NMC about the evader, along with arbitrary evader performance scale factors. Problem B will extend the linear regression model to include cross-track motion. Finally, Problem C will conduct a parameter study of a common RPO scenario involving a pursuer in an initial relative NMC orbit about an evader. Results for both the 2D and 3D case will be shown using the developed linear regression models. The parameter study will examine the effects of varying the NMC semi-major axis, the covariance of the pursuer's uncertainty ellipsoid, the performance scale factors, and pursuer starting angle relative to the evader to identify trends that can be used to form "rules of thumb" for spacecraft pursuit-evasion differential games. Problem D will seek to answer questions relating to the applicability of the developed linear regression models and IHM-PSO solutions. A sample of trajectories generated by the IHM-PSO method will be used as a guidance trajectory in a closed-loop Guidance Navigation and Control (GNC) algorithm to determine if the trajectories are viable in a closed-loop manner and in the presence of noisy measurements. Problem D will also take the planned control profile into high fidelity propagators and examine the open loop reference trajectory against realistically propagated results in order to gauge the level of inaccuracy inherent in using linearized dynamics without perturbations.

3.1 Problem A: 2D Linear Regression Model

In order to build the linear regression model, one first needs a collection of data. To solve the pursuit-evasion optimal control problem with the addition of uncertainty in starting position, the particle swarm function within the Matlab global optimization toolbox will be used with a script developed by Prince et. al [27], that was based off the indirect metaheustric algorithm devised by Pontani and Conway [25]. Pontanti and Conway state that for a zero-sum game with homogeneity of the costate equations and free final time, the optimal control transversality condition is ignorable. This implies that only the relative magnitude of the pursuer and evader costates must be satisfied at the boundary conditions [57]. Stupik [19] provides the costates for an intercept pursuit-evasion game as:

$$\begin{aligned} \lambda_1 + \lambda_7 = 0 \qquad & \lambda_2 + \lambda_8 = 0 \\ \lambda_3 + \lambda_9 = 0 \qquad & \lambda_4 + \lambda_{10} = 0 \\ \lambda_5 + \lambda_{11} = 0 \qquad & \lambda_6 + \lambda_{12} = 0 \end{aligned} \tag{3.1}$$

Prince [6] shows how the terminal constraints, ϕ, can be solved with Matlab's PSO algorithm. Whereas Stupik [19] formulates the problem with the initial costates, Prince et al. [27] formulate the same problem with the final costates. These are given by [27]:

$$\begin{aligned} \lambda_{1,f} = \nu_1 \qquad & \lambda_{7,f} = -\nu_1 \\ \lambda_{2,f} = \nu_2 \qquad & \lambda_{8,f} = -\nu_2 \\ \lambda_{3,f} = \nu_3 \qquad & \lambda_{9,f} = -\nu_3 \\ \lambda_{4,f} = 0 \qquad & \lambda_{10,f} = 0 \\ \lambda_{5,f} = 0 \qquad & \lambda_{11,f} = 0 \\ \lambda_{6,f} = 0 \qquad & \lambda_{12,f} = 0 \end{aligned} \tag{3.2}$$

where the ν_i values are the unknown terminal costates needed to enforce Equation (3.1) [19].

This research will leverage code previously developed by Prince et al. [27], which utilizes the Tschauner-Hempel equations of motion to handle eccentric chief reference orbits. For this research effort, the eccentricity of the chief orbit is set to zero, reducing the equations down to HCW dynamics. In Prince et al.'s [27] work, the utilization of the Tschauner-Hempel equations allow the Yamanaka-Ankersen [13] state transition matrix to propagate the states in terms of true anomaly, f. The Yamanaka Ankersen [13] matrix is then used by Prince et al. [27] to develop a state transition matrix for the costate equations:

$$[\Xi](f, f_0) = \left([\Theta(f, f_0)]^{-1}\right)^T \tag{3.3}$$

where $[\Xi]$ is the costate STM and $[\Theta]$ is the Yamanaka Ankersen STM. Following the Prince et al. solution process [27], shown graphically in Figure 10, the indirect heuristic method starts at the final state and propagates backward in time to the initial state in order to get the costates at the initial time, t_0. The states are then numerically propagated forward from the initial time, t_0, to the final time using both STMs. As a post processing step, the results are then converted back to the time domain. The control for the pursuer and evader, $\bar{u}$ and $\bar{\nu}$ respectively, is formulated as a continuous thrust able to be pointed in any direction, found by an azimuth and elevation angle. Therefore they can be written [6]

$$\begin{aligned} \bar{u} &= [\alpha_P, \phi_P]^T \\ \bar{\nu} &= [\alpha_E, \phi_E]^T \end{aligned} \tag{3.4}$$

Note that Equation (3.4) involves only thrust angles and does not contain a parameter for the magnitude of the thrust value. The magnitude of thrust is assumed continuously applied in the azimuth-elevation angles specified by the control law. The

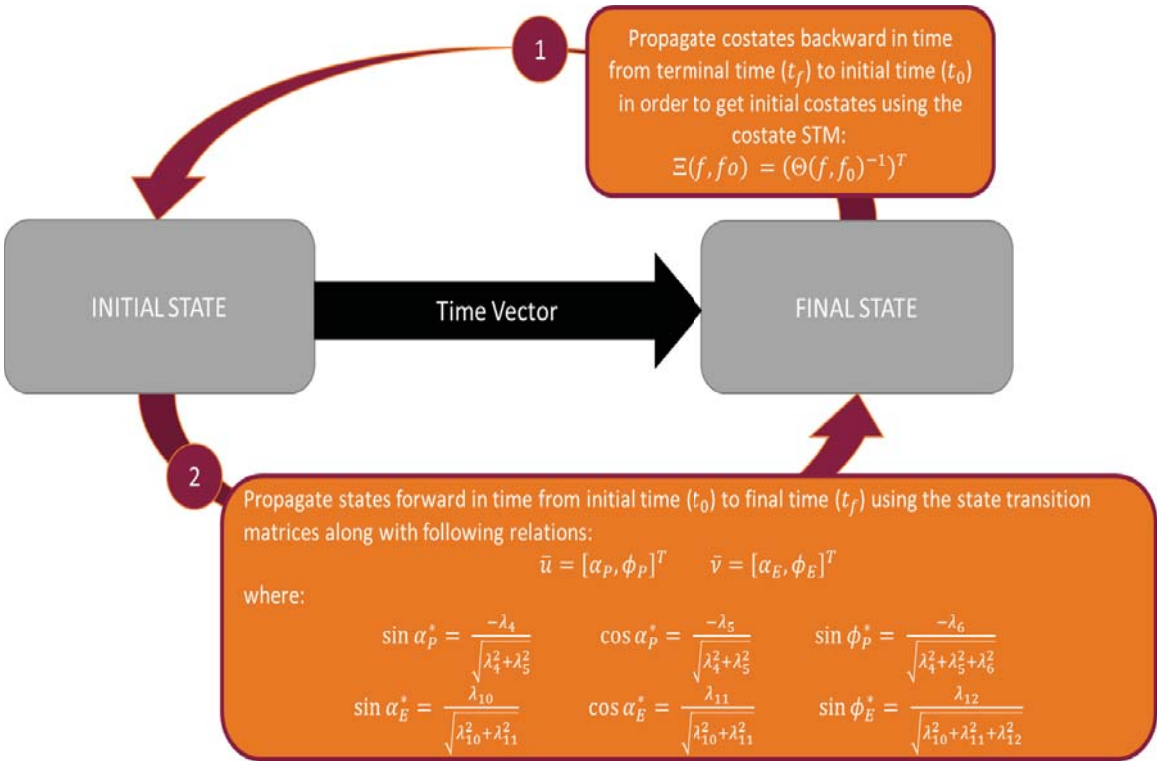


Figure 10. Indirect Heuristic Method Flow, adapted from [6]

magnitude of the applied thrust is directly affected by the performance scale factor and propellant mass loss given by Equation (3.5), taken from Prince [6]:

$$a(t) = \left(\frac{a_0}{1 - ta_0/c}\right) SF \tag{3.5}$$

where the initial control acceleration (a_0) is 3.43×10^{-5} km/sec^2, and the effective exhaust velocity (c) is 3 km/sec to match the work done by Prince et al. [27], Spendel [58], and Stupik [19]. These control angles can be solved for in relation to the costates to represent optimal steering laws [6]:

$$\begin{aligned} \sin\alpha_P^* &= \frac{-\lambda_4}{\sqrt{\lambda_4^2+\lambda_5^2}} \quad \cos\alpha_P^* = \frac{-\lambda_5}{\sqrt{\lambda_4^2+\lambda_5^2}} \quad \sin\phi_P^* = \frac{-\lambda_6}{\sqrt{\lambda_4^2+\lambda_5^2+\lambda_6^2}} \\ \sin\alpha_E^* &= \frac{\lambda_{10}}{\sqrt{\lambda_{10}^2+\lambda_{11}^2}} \quad \cos\alpha_E^* = \frac{\lambda_{11}}{\sqrt{\lambda_{10}^2+\lambda_{11}^2}} \quad \sin\phi_E^* = \frac{\lambda_{12}}{\sqrt{\lambda_{10}^2+\lambda_{11}^2+\lambda_{12}^2}} \end{aligned} \tag{3.6}$$

Prince et al. [27] show how the indirect heuristic method [25] transforms the optimal control problem into a type of parameter optimization problem, sometimes referred to as a boundary value problem. Matlab's particle swarm routine, within the Global Optimization Toolbox, is utilized with default options (except for an increase in max iteration limits), to solve the parameter optimization problem by obtaining the vector $\bar{\psi}$ of parameters [6]:

$$\bar{\psi} = \begin{bmatrix} \lambda_{1,f} - \lambda_{7,f} \\ \lambda_{2,f} - \lambda_{8,f} \\ \lambda_{3,f} - \lambda_{9,f} \end{bmatrix} \tag{3.7}$$

Given that the terminal costates are equal and opposite for the pursuer and evader, $\bar{\psi}$ should be zero. However, the tendency of the PSO to return sub-optimal results is quite high, and thus the cost function to minimize in the PSO process is [6]:

$$J_{PSO} = \sum_{i}^{m=3} \left(\bar{\psi}(i)^2 \right) \tag{3.8}$$

3.1.1 Application of IHM-PSO Algorithm.

Given a single set of initial conditions consisting of a starting position for the pursuer and evader, the IHM-PSO routine [25, 27] is able to solve the two-player pursuit-evasion differential game. The next step is to build a collection of data for the linear regression model, as is outlined in Figure 11. The inherited code from Prince et al. [27], has been modified to work as a single function for this research. All of the separate function files were compiled into a single main function that takes initial conditions of the game as inputs, and outputs the solved optimal control trajectory as an output. This makes deploying the function possible on multiple computers without having to deal with search paths and callbacks. One layer above the optimal control solver is a script, called the "intermediate script" in Figure 11, which iteratively

executes each optimal control simulation. After each run, the intermediate function stores the information and provides the optimal control solver with the next pursuer starting position. During this process, the intermediate layer plots each completed simulation result so that one can monitor progress as the simulation goes on. The highest layer of the program, called the "Control Loop Script" in Figure 11, controls the variation in parameters such as scale factor, pursuer relative phasing angle, etc. For building the linear regression model, the control loop script will be fairly simplistic as the only change being made is the evader performance scale factor. The other function of the control loop script is to save individual runs to their designated file folder, labeling the files appropriately to avoid the intermediate layer overwriting successive runs as the factors such as evader performance scale factors are varied.

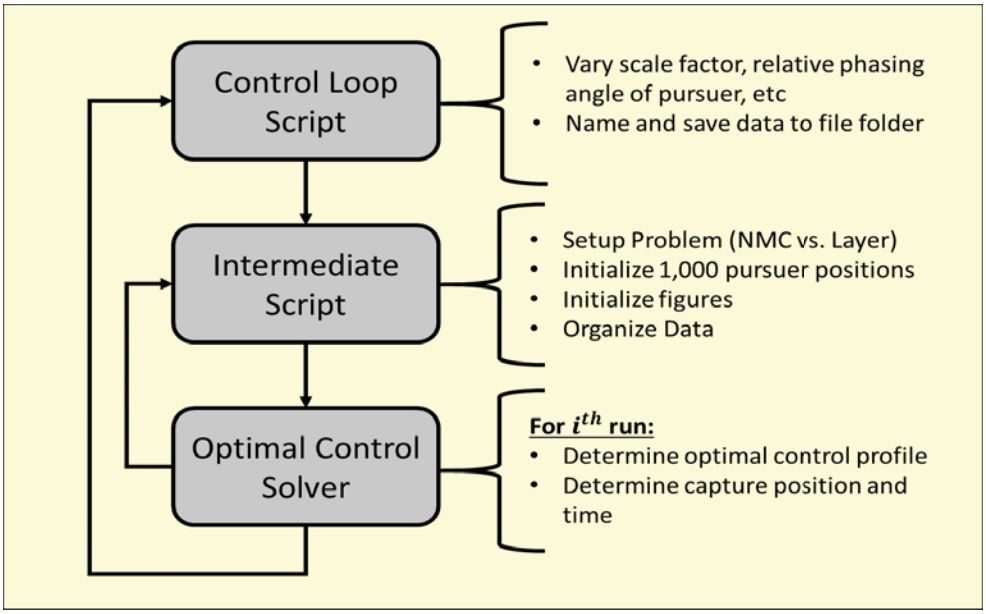


Figure 11. Programming Overview

To build the linear regression model, the intermediate script will setup a series of pursuer positions that evenly sample the total solution space. For the 2D (in-plane) case a 50×50 km square sampled at intervals of 2 km for a total of 2601 pursuer positions, shown in Figure 12. For the 3D case (where cross-track motion is allowed),

the $50 \times 50 \times 50$ km cube centered at the origin will be sampled at an interval of 7 km for a total of 3375 pursuer positions. Note that these variations are all bounded NMC orbits about the evader (at in the LVLH frame), and the chief reference orbit is a geosynchronous equatorial orbit. At each of these pursuer positions, the IHM-PSO solver will determine an optimal control profile, capture position, and capture time. The intermediate program will store all the data produced by the IHM-PSO solver to a ".mat" file every so often as the loop iteration progresses. In this way, the data is stored as the program is executing. If the program is interrupted before it finishes, the loop can be restarted from the last checkpoint simply by changing a declaration variable at the top of the intermediate script and running the program again.

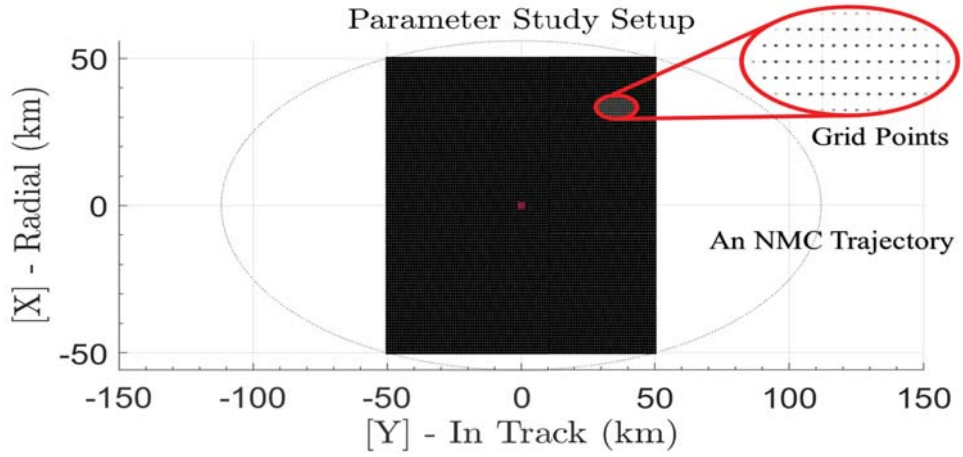
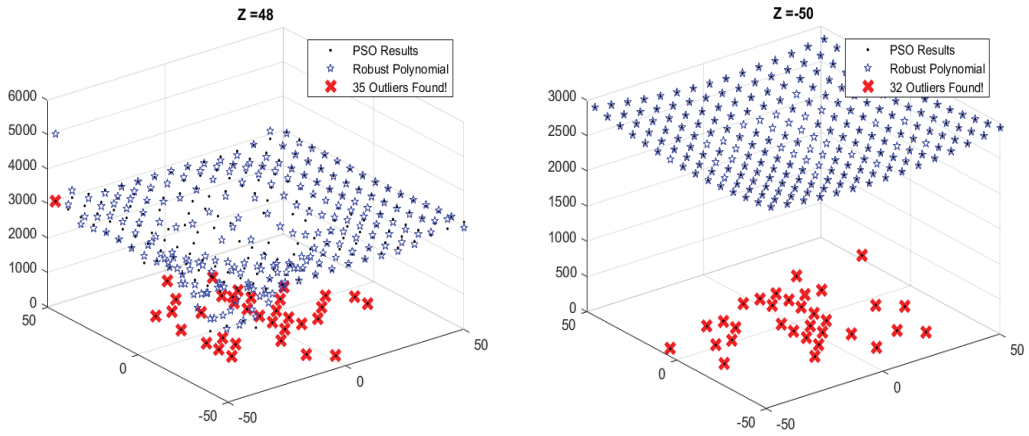


Figure 12. Set-up for the Parameter Study Analysis (Note only one of many NMC trajectories is shown)

3.1.2 Outlier Identification and Removal.

Prince et al. [27] states the developed PSO algorithm usually provides accurate optimal results 70-90% of the time. During the process of building the linear regression models this statistic was largely verified, however it was noted that the lower bound on the stated accuracy could be much lower for certain repeatable circumstances like those shown in Figure 13. In these instances, the PSO particles are believed to be

too strongly attracted to a sub-optimal local minimum while searching the solution space. A solution to this problem would be to tune the social and cognitive weights, however doing so would invalidate the other data runs for comparison and use in the same linear regression model.



(a) **(b)**

Figure 13. IHM-PSO Results with Rampant Outliers

In more common scenarios, only a couple of outliers were found to be present, which means that the results from the constructed grid, shown in Figure 12, will almost certainly contain some sub-optimal results. These sub-optimal results will negatively impact the accuracy of the linear regression model and must be dealt with. Either the data will need to be cleaned to remove the outliers, or the method of constructing the linear regression model must be able to identify and ignore outliers. Where possible, Matlab's robust linear regression toolbox will be used to build the linear regression models and at the same time ignore the suboptimal results in the IHM-PSO results. When Matlab's robust programs cannot be used, then two other methods are used to identify outliers in the data set. Once identified, they can be either removed or the IHM-PSO algorithm [27] can be rerun at those selected points.

By taking advantage of the symmetry of the capture position's solution set shown in Figure 14, potential outliers can be determined. First the median of the data set is found, and then the second half of the data is flipped and compared to the first half. At each point the data is compared, and correspondingly large discrepancies between the two symmetric points identify possible outliers. It is important to note that this method requires no points to have been removed prior, because doing so breaks the symmetry of the data.

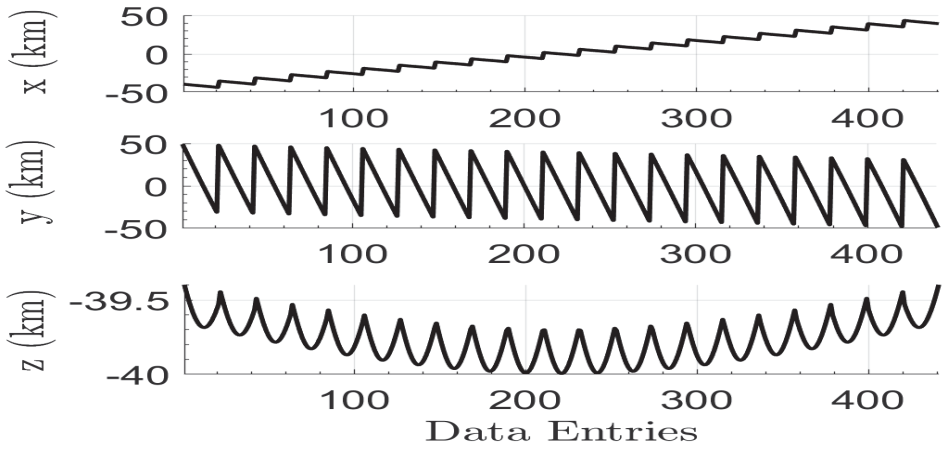


Figure 14. Symmetry of Capture Data Set

Another tool used to find outliers within a data set is the Matlab built-in function "isoutlier", using the method "movmedian". This algorithm moves across the data set with a small window. Within this small subsection of the data the median is utilized to identify outliers. The "isoutlier" routine returns data indices of values found to be 3 standard deviations (σ) from the local median [59]. This method is largely accurate, however sensitive to the user defined window size. When given too large of a window, the method fails to report all outliers. With a too small window, one bad outlier affects the local median too much. Figure 15 shows an example of the moving median method not finding all of the outliers with the given 3σ range.

Shrinking the 3σ range often finds the outliers better, but shrinking the confidence interval too much has a tendency to return false positives.

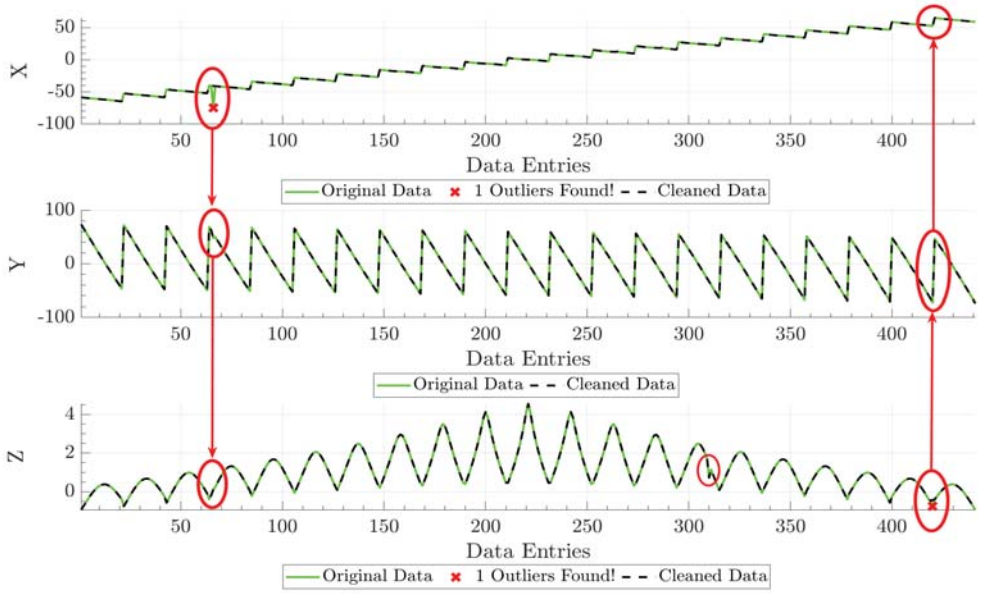


Figure 15. Using Matlab's Moving Median Outlier Identification Method [59]

The above methods largely automate the data cleaning process, however they were intentionally not fully automated. The author checked each time the methods completed in order to visually identify any outliers that the methods missed. For example the moving median method sometimes fails to find outliers that are direct neighbors because the local median is too far skewed. Other times it was observed that both points selected in the mirror method were suboptimal, and a false negative was given as a result.

3.1.3 Linear Regression.

With a large collection of clean data, one can formulate an input-output empirical relation with regression techniques. Looking at the process shown in Figure 16, the mapping can be thought of as a multi-input multi-output (MIMO) system.

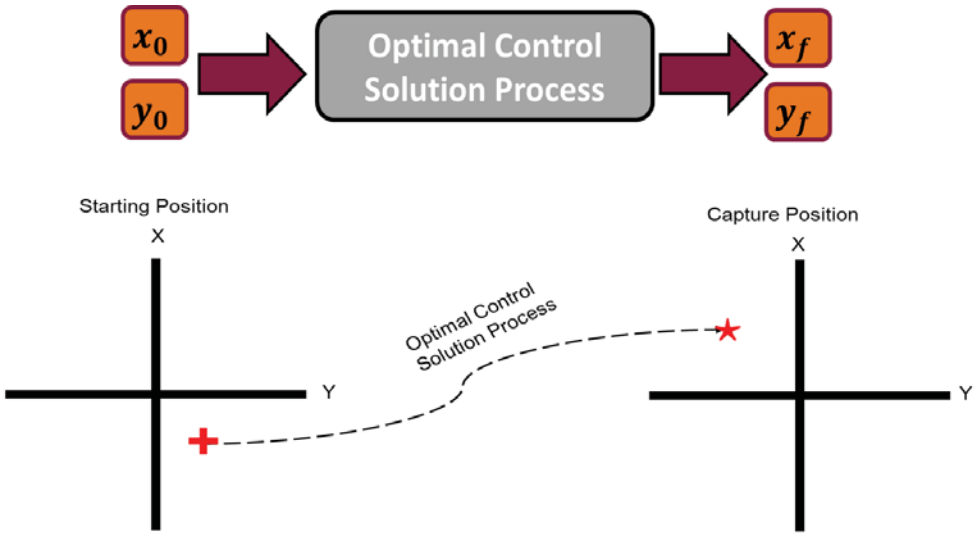


Figure 16. Optimal Control Mapping Process

Matlab's robust multivariate regression tools are setup to handle only multi-input, single-output (MISO) systems. Fortunately, the MIMO system shown in Figure 16 can be decomposed into two MISO systems and analyzed separately. This does not mean that the radial and in-track positions are decoupled, but merely that the mapping to final positions can be done separately. A linear regression model is then created for each output ($\bar{x}_f$ and $\bar{y}_f$), which takes in both pursuer initial state inputs ($\bar{x}_0$ and $\bar{y}_0$).

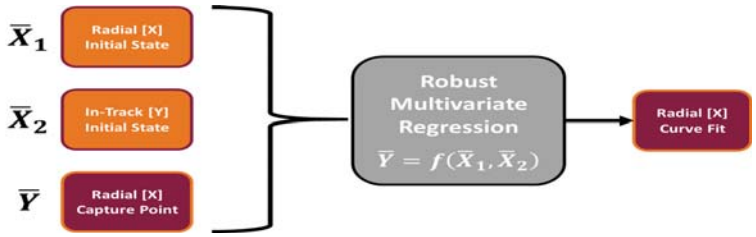


Figure 17. Robust Multivariate Regression Mapping Process

The result is a pair of functions, shown in Figure 18, that together take in the initial pursuer positions and output the estimated capture position. Through experimentation, a seventh-order polynomial fit was found to produce the best results (see Table 8, 9, and 10).

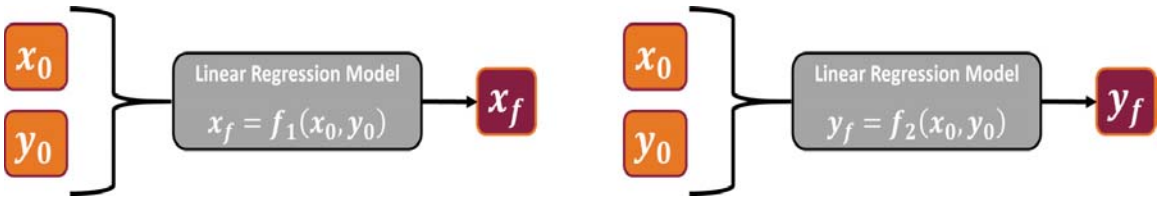


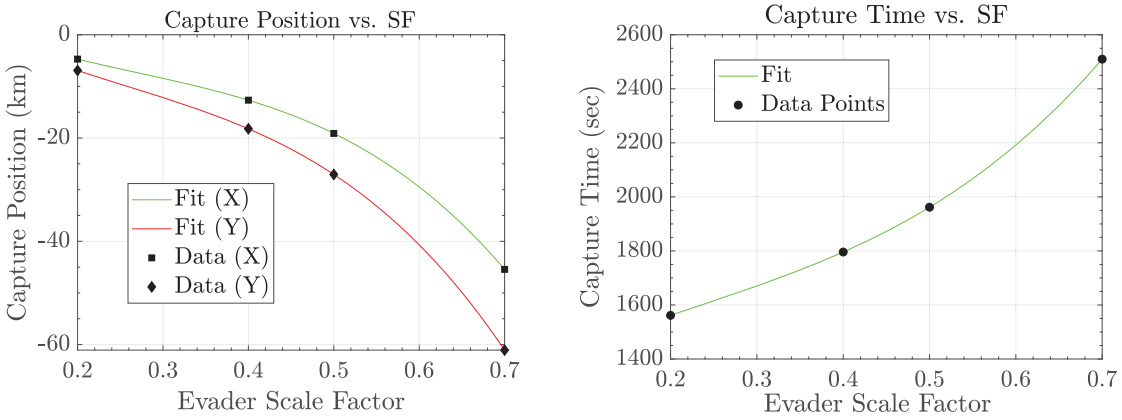
Figure 18. Linear Regression Equations

The evader performance scale factor was not an input into the regression curve fit process, and was instead kept as a given constant inherent as an assumption in the optimal control solution process. The evader performance scale factor, SF, varies the available control acceleration of the evader relative to the pursuer:

$$a_{0,e} = \text{SF } a_{0,p} \tag{3.9}$$

where $a_{0,p}$ is 3.43×10^{-5} km/sec^2 and SF ranges from 0.05 and 0.70. To further improve the linear regression model, the capability for arbitrary evader performance scale factors is addressed. This is accomplished by evaluating each linear regression model at the given initial condition over a variety of evader performance scale factors, and then performing a curve fit on the collection of points, as shown in Figure 19. Thus, the linear regression model is able to take arbitrary initial pursuer positions as well as arbitrary evader performance scale factors as an input to produce the expected capture position. It is important to note that to ensure accuracy of the linear regression model, the inputs should be within, or near, the range of data used to create the model. This research effort utilizes HCW dynamics that start to lose accuracy as the distances from the virtual chief (the origin of the LVLH frame) increase. Thus for a GEO scenario, the linear model was built from $-50 \leq x \leq 50$ km, $-50 \leq y_0 \leq 50$ km, and evader performance scale factors ranging from 0.05 to 0.7 at intervals of 0.05. The 50 km square and max evader performance scale factor of 0.7 were chosen to ensure that the capture positions found with the IHM-PSO algorithm

do not exceed more than approximately 100km from the origin of the LVLH frame. Note, that for the case shown in Figure 19, the maximum capture time is only 3% the orbital period, and the maximum capture position is 0.14% the orbital chief radius, thus satisfying the assumptions and limitations outlined in Section **??**.

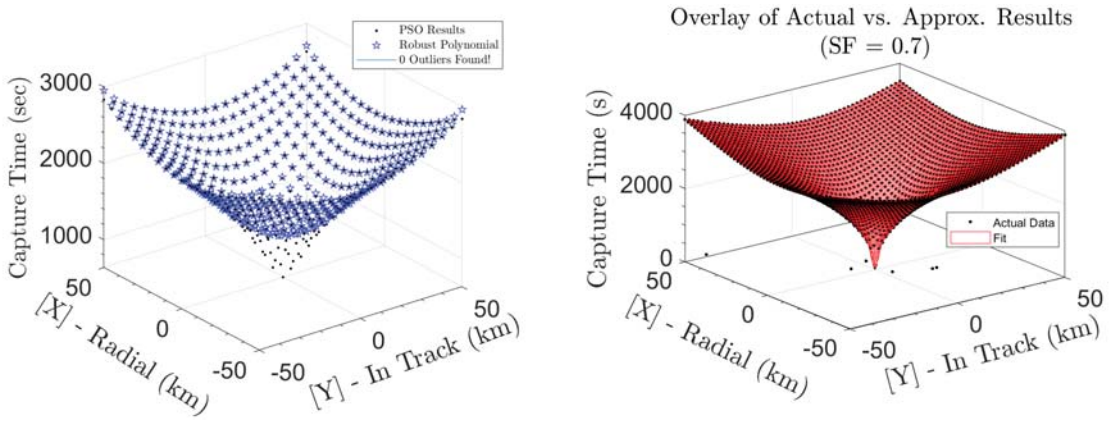


(a) Variance in Capture Position vs. SF **(b) Variance in Capture Time vs SF**

Figure 19. Variance as a Function of Scale Factor

Along with the capture position data, the IHM-PSO method also provides the capture time. Shown in Figure 20b, an example of the capture time data points from the PSO solution is plotted with respect to initial pursuer position. The complexity of the solution's surface can not adequately be fit by any expression within the Matlab regression toolbox library because of the funnel shape that constricts down to the origin. Trying to fit this surface with a polynomial approximation, shown in Figure 20a, produces large errors when the polynomial approximation fails to reach all the way down the funnel. This inaccuracy precludes the use of the linear regression curve fitting process that was applied to the capture position data. Instead, an interpolation of data is performed using the Matlab function "fit()" with the method "nearestinterp." This process uses linear interpolation on the given data using the nearest neighbor approach [60], which means two things. First, interpolation inher-

ently has an accuracy proportional to the density of the data sampled. And second, the interpolation routine does not have a robust option like the linear regression toolbox does, which then implies that the erroneous results contained within the data set will significantly affect the interpolation process. Thus the methods described for locating and eliminating outliers in Section 3.1.2 will prove crucial for ensuring the accuracy of the capture time data fits.



(a) Polynomial Approximate Example **(b) Linear Interpolation Example**

Figure 20. Solution Set Comparison

3.2 Problem B: 3D Linear Regression Model

3.2.1 Extending Linear Regression Model to 3D.

Incorporation of cross-track motion, even though it is a simple harmonic oscillator [14], presents several challenges by virtue of adding an additional degree of freedom to the problem. Lovell's relative orbital elements [14] will again be used to shape the desired relative NMC and find the initial velocities required for the pursuer. To determine the one additional cross-track velocity component requires two additional Lovell relative orbital element parameters [14], ψ and z_0. The required initial cross-

track velocity, $\dot{z}$, is found by [14]:

$$\dot{z}_0 = \frac{z_0 n}{\tan(\psi_0)} \tag{3.10}$$

where ψ_0 is the initial angle of the cross-track motion. From Equation (3.10), there exists a singularity if $\tan(\psi_0)$ is allowed to be zero. Thus for the parameter study and linear model, $\tan(\psi_0)$ will be purposely chosen to be nonzero.

In the context of this problem there are four parameters that define the 3D orbit: the three relative Cartesian coordinates (x_0,y_0,z_0) and ψ_0. Going from two parameters to four substantially increases the number of layers required to create the linear regression model since each combination of cross-track angle (ψ_0) and evader performance SF are needed, in addition to extra layers needed for z_0. Thus in order to reduce the number of total iterations, the parameters that make up the linear regression model were spaced more loosely than in the 2D model. The 50km cube of points that the IHM-PSO algorithm is evaluated on was spaced at an interval of 7 km. The evader performance SF ranged from 0.10 to 0.70 in increments of 0.10. And finally the cross-track angle was spaced at angles: $5°, 50°, 85°, 95°, 130°$, and $165°$. A total of 3,375 points were used to evaluate the $50 \times 50 \times 50$ km cube for each variation of ψ_0 and SF.

3.2.2 3D Regression Model.

At first attempt, the linear regression model was built by including the cross-track component as a third independent variable, $\bar{X}_3$ in the linear regression formula:

$$\bar{Y} = f(\bar{X}_1, \bar{X}_2, \bar{X}_3) \tag{3.11}$$

where $\bar{X}_1$ represents the initial radial positions, and $\bar{X}_2$ represents the initial in-track positions, and $\bar{X}_3$ represents the initial cross-track positions. While Matlab was successful at building a linear regression model, the accuracy was poor. Experimentation among the various regression models failed to improve the accuracy of the linear model. Further examination found that there were small "waves" within the solution set, shown in Figure 21, that appeared to cause the regression algorithm trouble as it tried to account for these variations. Furthermore it can be seen that the regression fit falls off near the corners of the grid, where accuracy deteriorates to as much as 3 km off.

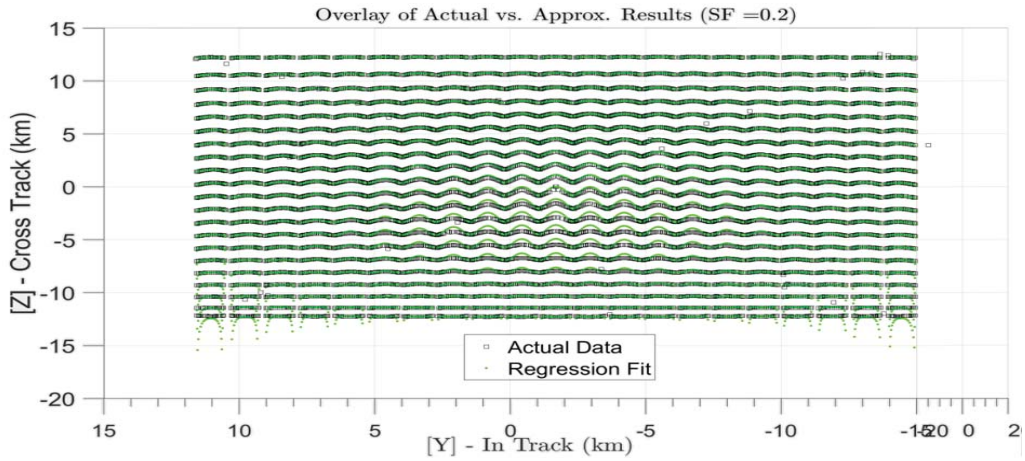
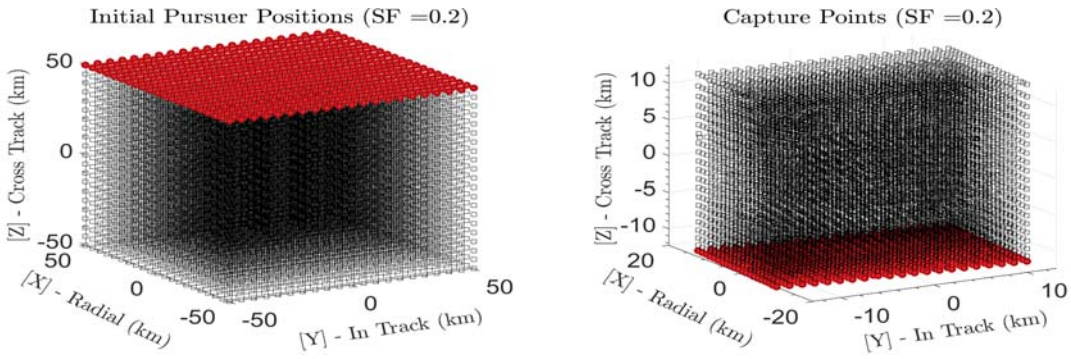


Figure 21. Matlab 3D Regression

Given that Matlab's linear regression was highly accurate in the 2D case, it was sought to use multiple 2D regression layers stacked by their associated Z height. The use of HCW equations allows the in-plane and cross-track motion to be separated, which allows the 3D linear regression model to become a stack of 2D linear regression models with variations in the initial cross-track starting positions. This point is shown in Figure 22, where all starting positions with a initial cross-track value of 50 km in the IHM-PSO data map to the same layer in the capture region (Figure 22b).



(a) Starting Positions at $z_0 = 50$ km **(b) Associated Capture Positions**

Figure 22. Z Height Separability of IHM-PSO Data

Thus the 3D linear regression model first needs to determine the correct z height (given by the initial pursuer starting position), and then evaluate the correct 2D linear regression model to get the final radial and in-track capture position. To determine the final cross-track capture position, a regression identical to that discussed in Section 3.1.3 is performed. When the initial z_0 condition does not coincide with any of the 2D regression layers, then the initial (x, y) pair are evaluated and fit across all of the z_0 layers. Figure 23 shows how the 3D regression model is built and evaluated for arbitrary inputs of $x_0, y_0, z_0, \psi_0,$ and SF.

Each 3D linear regression model was run with a given constant value for evader performance SF and initial cross-track angle, ψ_0. If a selection of initial pursuer cross-track angle and evader performance SF values are chosen coincident with a particular model, then only that model is needed to evaluate the answer. More than likely this will not be the case, and the initial values will require interpolation between linear model combinations, as depicted in Figure 23. Thus in summary the 3D linear regression model is a collection of 2D linear regression models run for variations of z_0 heights, evader performance SFs, and pursuer initial cross-track phase angles. The groups of linear models are interpolated between using bilinear interpolation to obtain

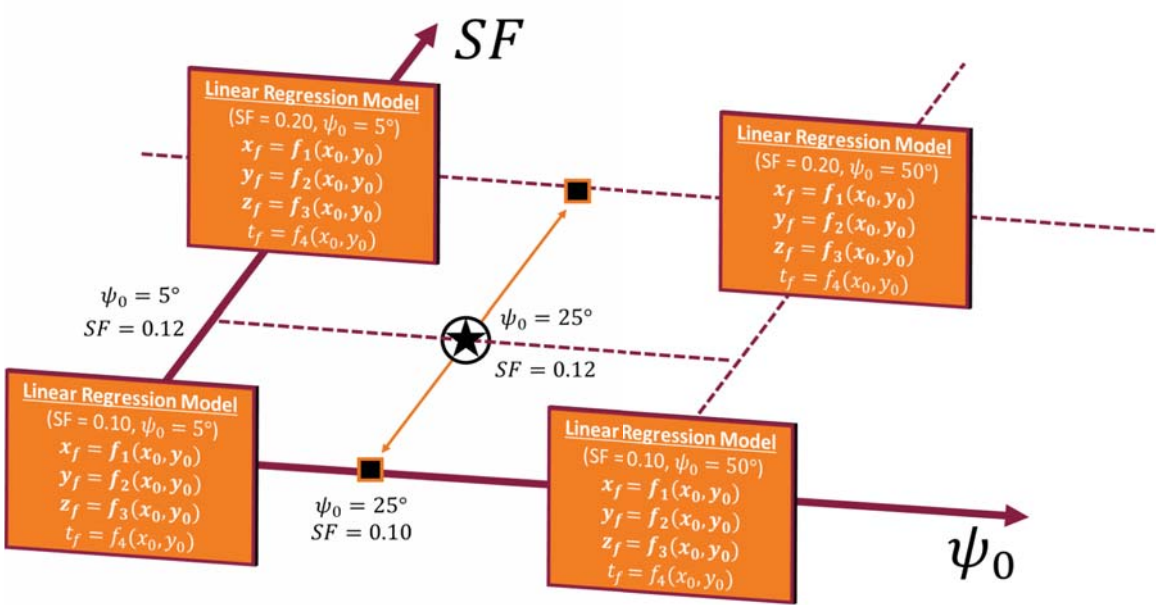


Figure 23. 3D Linear Regression

values not explicitly evaluated at, which provides an means for obtaining results for arbitrary initial conditions.

3.3 Problem C: 2D and 3D Parameter Study

3.3.1 2D Parameter Study.

This section focuses on applying the linear regression models developed in Problem A (Section 3.1) and Problem B (Section 3.2) to analyze the behavior of the solution set when uncertainty is applied to the pursuer's initial starting position relative to the evader's within the orbital plane, as shown in Figure 24. Note: this implies that the evader always starts at the origin of the LVLH frame. The initial radial and in-track velocity of the pursuer is initialized such that the pursuer is in a NMC trajectory about the origin (and the evader). For the 2D analysis the cross-track initial velocity is set to zero, effectively ensuring that the game is played out in-plane. To compare various positions of the pursuer relative to the evader, the pursuer's position is varied

such that the initial position remains on the same NMC, but at different phasing angles (see Figure 5) from the evader. A covariance ellipse, centered on each of the chosen pursuer's positions, is drawn based upon a fixed confidence interval of 95% (2 standard deviations) utilizing code developed by Johnson [61]. A random sampling of 1,000 normally distributed points are taken within the covariance ellipse, which then represent all of the initial pursuer starting positions that need to be mapped to final capture positions. Running the PSO-IHM [27] algorithm at each of these starting positions would take approximately 32 hours per scale factor. To do a full run across the seven evader performance scale factors would then take 9 days, and 8 hours. The wait time of 9 days and 8 hours would have to be repeated for each variation on the NMC semi-major axis and initial covariances. Conversely, the linear regression model produced estimates of the capture time and position for the entire parameter study in a matter of 30 minutes or less.

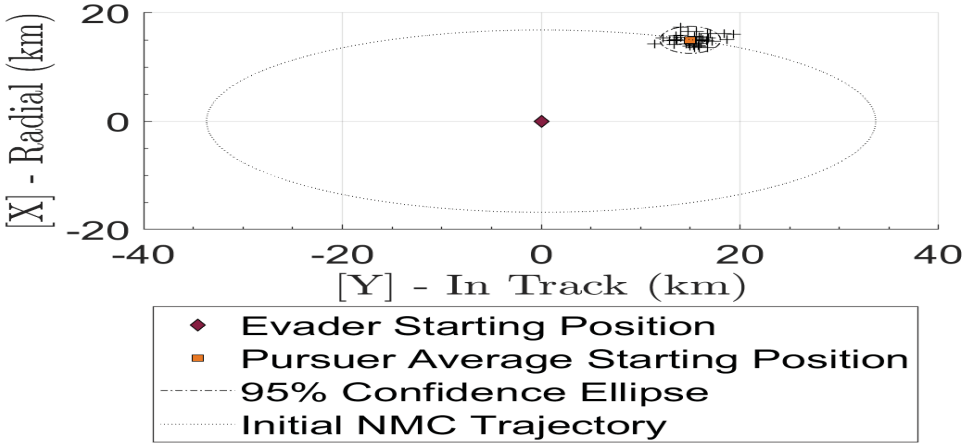


Figure 24. Set-up for the 2D Analysis

3.3.2 3D Parameter Study.

Similar to the 2D parameter study, for the 3D parameter study it is desired to provide relative orbital angles instead of Cartesian coordinates so that the pursuer starting angles can be varied while keeping semi-major axis of the relative orbit the

same. To solve for the initial cross-track velocity, and initial cross-track position, some more work with Lovell's relative orbital elements [14] is needed. From Lovell [14], let the maximum cross-track amplitude A_z and initial cross-track angle, ψ_0 be given:

$$\psi_0 = \text{atan2}\left(z_0, \frac{\dot{z}_0}{n}\right) \tag{3.12}$$

$$A_z = \sqrt{z_0^2 + \left(\frac{\dot{z}_0}{n}\right)^2} \tag{3.13}$$

where the initial cross-track position, z_0 and cross-track velocity, $\dot{z}_0$ are unknown. Rearranging the equations and solving for z_0 yields:

$$z_0 = \frac{A_z \tan(\psi_0)}{\sqrt{\tan^2(\psi_0) + 1}} \tag{3.14}$$

where $\dot{z}_0$ can be found by rearranging Equation (3.13):

$$\dot{z}_0 = n\sqrt{A_z^2 - z_0^2} \tag{3.15}$$

Note that this method for solving for $\dot{z}_0$ was judiciously chosen since an answer using Equation (3.12) would contain a singularity when $\psi_0 = 2\pi k$ where $k = 0, 1, 2, ...\infty$.

As a result, in the 3D parameter study the pursuer is located on a initial relative NMC about the origin and the evader by $\nu_{r,0}$, ψ_0, a_r, and A_z. As seen in Figure 25, varying the relative phasing angle ν_r varies the angle that the pursuer starts at very similarly to the true anomaly orbital element. Variations to ψ_0 have the effect of changing the initial z height at the starting angle found by ν_r. A change to A_z affects the maximum value of cross-track motion. And finally, a_r changes affect the semi-major axis of the 2×1 relative orbit ellipse.

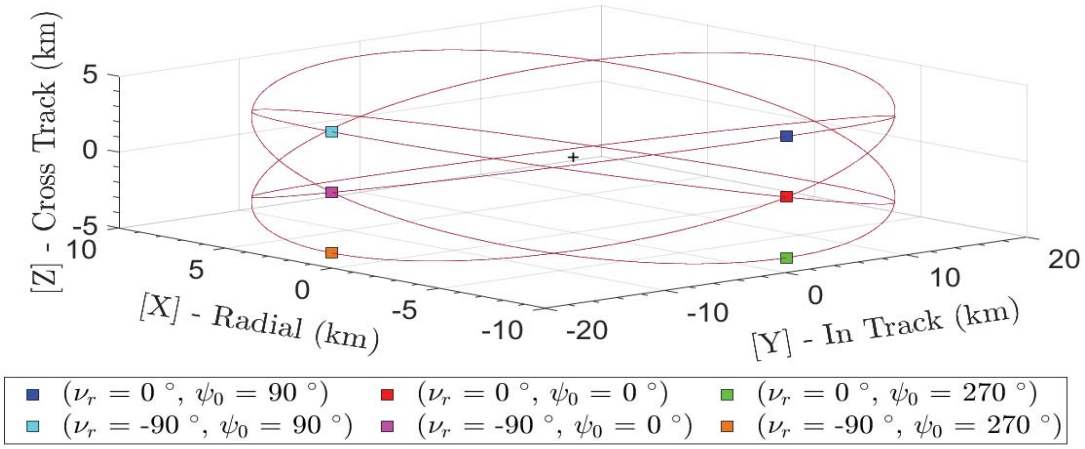


Figure 25. Moving Around in the 3D Relative Orbit

As a result of reworking the Lovell relative orbital elements [14] the 3D parameter study can be conducted by varying the relative pursuer starting angles, relative orbit semi-major axis, covariance ellipse sizes, and maximum cross-track amplitude. The setup for this parameter study is shown in Figure 26.

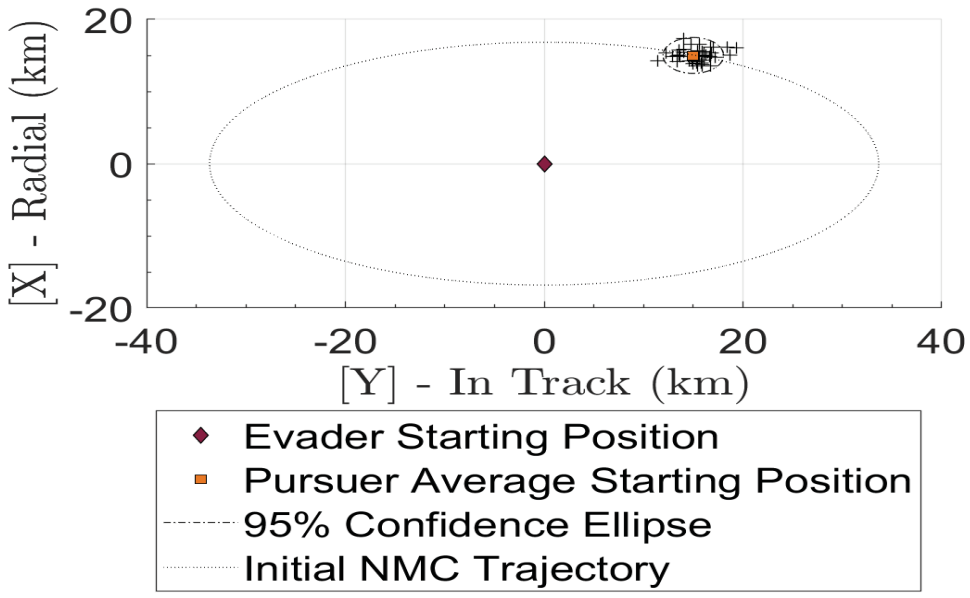


Figure 26. Set-up for the 3D Analysis

3.4 Problem D: Closed-Loop Control and High Fidelity Propagation

3.4.1 GNC Architecture.

This problem takes the open-loop guidance trajectories generated in Problem A and Problem B, and applies them in a closed-loop manner with a developed GNC model [45]. The GNC system is modeled in Matlab's Simulink, and consists of the guidance, navigation, control, plant, and sensors subsystems as shown in Figure 27.

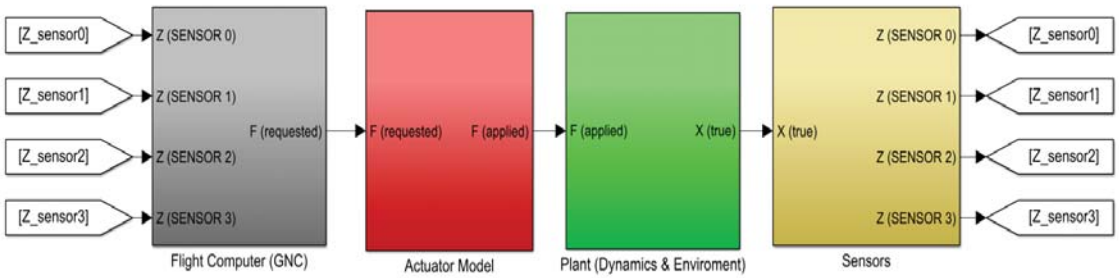


Figure 27. Main Layer of GNC Simulation

The GNC model is capable of using a linear Kalman filter, extended Kalman filter, unscented Kalman filter, or particle filter. Harris et al. [45] found that the LKF was sufficient for most applications (dependent upon the sensor and filter combination used) and easiest to implement. Thus it will be the filter chosen for this application. Using the developed Matlab Appdesigner front end [45], as shown in Figure 28, easily allowed various sensor/filter combinations to be run. The LKF and LQR choices were chosen within the GUI after implementing all other combinations, showing the power and flexibility of the modular GNC architecture.

The flight computer component, shown in gray in Figure 27, consists of a navigation and controller subsystems. The navigation subsystem determines where the spacecraft is in relation to the reference trajectory by using one of a variety of filters. Within the flight computer is also a controller, which takes the information provided by the navigation subsystem and determines the required thrust input to drive the

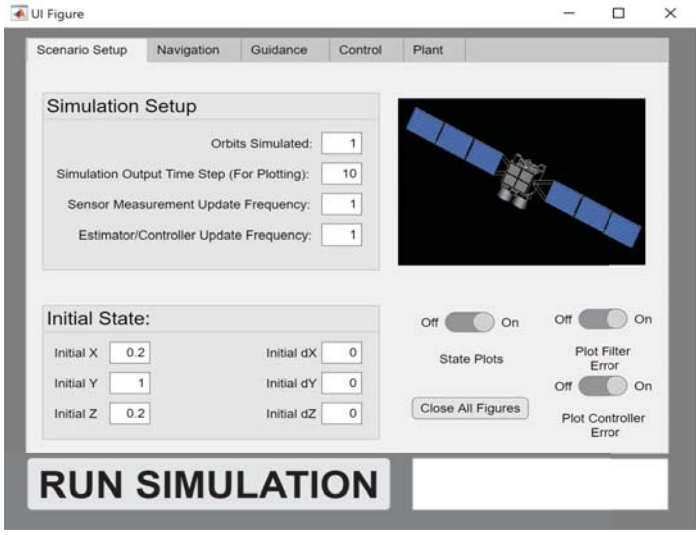


Figure 28. GNC GUI App

error between the actual and reference measurements to zero. Thus the open-loop guidance trajectories developed in Problems A and B will be differenced with the navigation measurements and then translated into thrust maneuvers within the flight computer block of Simulink. The control input will be calculated with a simple linear quadratic controller using negative feedback control, where the LQR gain K obtained from the steady-state algebraic Riccati equation. While the GNC is capable of more complex types of control, the LQR was found to be most robust across a wide variety of initial conditions, and much easier to tune than the other control methods that are available in the GNC system.

The actuator component in Figure 27 is responsible for interpreting the flight controller's requested amount of thrust and firing the thrusters accordingly. This is the component responsible for modeling effects such as thruster pointing inaccuracies, time delays in thruster firing, etc. For the purposes of this research, the actuator block will be setup to model a single low thrust electric propulsion engine. Mass loss will be accounted for matching the model used in the IHM-PSO algorithm [6], given in Equation (3.5), which provides the magnitude of the control acceleration. In the

actuator block, this term is calculated and represents the total magnitude of acceleration available from the thruster, $a_{available}$. The controller's requested acceleration, $\bar{a}_{requested}$, is then saturated if $||\bar{a}_{requested}|| > a_{available}$. Otherwise, it is left the same and $a_{available} = ||\bar{a}_{requested}||$. Using the following relation, the applied acceleration, $\bar{a}_{applied}$, given to the plant is:

$$\bar{a}_{applied} = \left(\frac{\bar{a}_{requested}}{||\bar{a}_{requested}||} \right) a_{available} \tag{3.16}$$

Essentially, the orientation of the thrust vector desired by the flight controller is preserved while the magnitude is scaled by the calculated control acceleration from the mass loss in Equation (3.5).

The plant block in Figure 27 propagates the provided control to the next time step by integrating the NERMs. Disturbance and perturbation forces are not accounted for in this formulation, however could easily be added by swapping out this dynamics block entirely, or by adding in perturbing effects like J_2 as extra components added to the acceleration terms.

Finally, the sensor component in Figure 27 takes the propagated states from the plant model and adds additive white Gaussian noise to simulate a noisy state. With the noisy state, the sensor model makes a measurement using either a range/range-rate, range, angles/range, or angles only sensor. The sensor measurements are returned to the flight computer in the next time step for the next cycle to repeat.

3.4.2 High Fidelity Propagation.

To gauge the total amount of inaccuracies inherent in the linearized set of dynamics used, AGI's STK software [62] will be utilized to propagate the generated IHM-PSO solutions with a high fidelity propagator called the High-Precision Orbit Propagator (HPOP). HPOP is capable of including perturbations such as Solar

Radiation Pressure (SRP), third body gravitational effects, and a full fidelity earth gravitational model [63]. Within STK, a mission planning aid called Astrogator is used to input impulsive maneuvers and coasting periods. Recall, the IHM-PSO solution trajectories are formulated in continuous thrust and thus will be converted from a continuous to impulsive thrust model. To make this conversion, code developed by LeValley and George [64] will be utilized. Using Matlab's unconstrained optimization tool "fminunc", the code creates an impulsive burn-coast-burn profile that matches the position and velocity of the given continuous trajectory using an HCW propagation model. Within the spacecraft parameters, both the evader and pursuer were left with identical default values. For each the engine model was generically selected as a constant I_{sp}, constant thrust model.

Astrogator is also capable of handling and propagating continuous thrust trajectories. Unfortunately STK is fairly rigid in the types of continuous thrust profiles that it can be given. To vary either the body or thrust axes an attitude ".a" file is required. This would require converting the Azimuth/Elevation angles into one of the required formats: Euler angles, quaternions, roll-pith-yaw angles, or ECI vector components. With the correct angle representations, one then has to write the angles into a specially formatted ASCII text file [65]. This route was attempted for comparison purposes, but unsuccessful since STK only recognized the propagation file but did not propagate it.

STK was driven with a Matlab script using the object model language. The script automatically creates the STK scenario, initializes the satellites parameters, and configures the burn lists. This is especially handy since a large number of burns can be used, and one can avoid having to manually configure the segments in the Astrogator GUI. A reference satellite is inserted into the GEO belt that establishes the origin of the LVLH coordinate frame. The pursuer and evader coordinate systems are

placed into a LVLH coordinate system of the reference satellite object. To extract the positional state information, custom vectors were created in the analysis workbench that measured distance from the pursuer and evader with respect to the LVLH origin as shown in Figure 29.

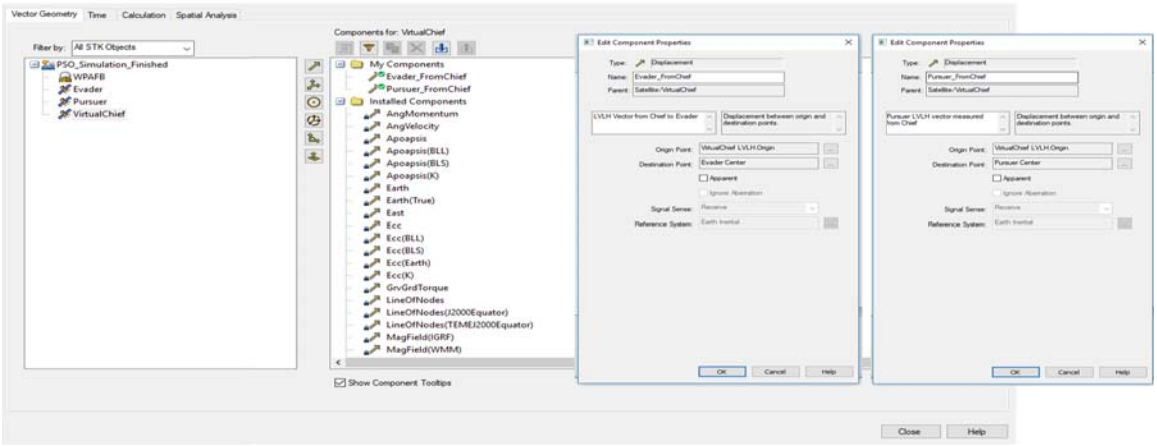


Figure 29. STK LVLH Vectors

3.4.3 Chapter Summary.

The methodology for each problem has been laid out in detail. Problem A showed how a large data set could be constructed by iteratively running an existing deterministic solver across a wide range of pursuer starting positions. Then, the setup for a 2D linear regression model was presented as a way to find empirical solutions to the data set. Problem B increased the complexity of the formulation by adding in cross-track motion, which required interpolation across nodes consisting of 2D linear regression models stacked by z height. with these the 2D and 3D linear regression models, Problem C setup a scenario to evaluate a large scale parameter study. Finally, Problem D discussed the developed GNC system and implementation of commercial software in order to validate the IHM-PSO solution trajectories accuracy and viability.

IV. Results

This chapter discusses the results for each problem. For Problem A (Section 3.1) the 2D linear regression model is shown to have extremely high accuracy ($< 0.01\%$) as a result of the robust multivariate regression techniques used to create it. Problem B (Section 3.2) successfully extends the linear regression model to three dimensions, but because of interpolation techniques, exhibits an average estimation error of 2%. Problem C (Section 4.3) applies the linear regression models to a 2D and 3D pursuit evasion game in order to develop "rules of thumb" from trends in the solution set for both cases. Finally, Problem D (Section 3.4) shows that the trends shown in Problem C persist in higher fidelity dynamics models, and that the accuracy from using linearized dynamics and ignoring perturbative effects is acceptable. It is also shown that the open loop trajectories are largely viable given sufficient acceleration limits.

4.1 Problem A: 2D Linear Regression Model

4.1.1 2D Linear Regression Model Results.

The linear regression model was evaluated at each of the pursuer initial starting positions on the grid in Figure 12 and then compared to the results obtained through the PSO algorithm. Shown in Figure 30 are the terminal capture positions estimated by the linear regression model overlaid upon the results found by the PSO algorithm. Figure 30 highlights areas where the PSO algorithm produces erroneous suboptimal results. At these problem areas, the model correctly predicts the correct capture point. If the PSO algorithm were to be run again at the points where suboptimal results were provided, the output would eventually converge to the points predicted by the model. In comparing the speed of these two algorithms, the PSO routine

produced results in approximately 8-12 hours, whereas the linear regression model completed the mapping in less than 3 minutes. Thus, when the optimal control profile is not needed and there is only interest in the terminal conditions, the linear regression model is an ideal tool for providing results on the same level of accuracy as that of the PSO algorithm.

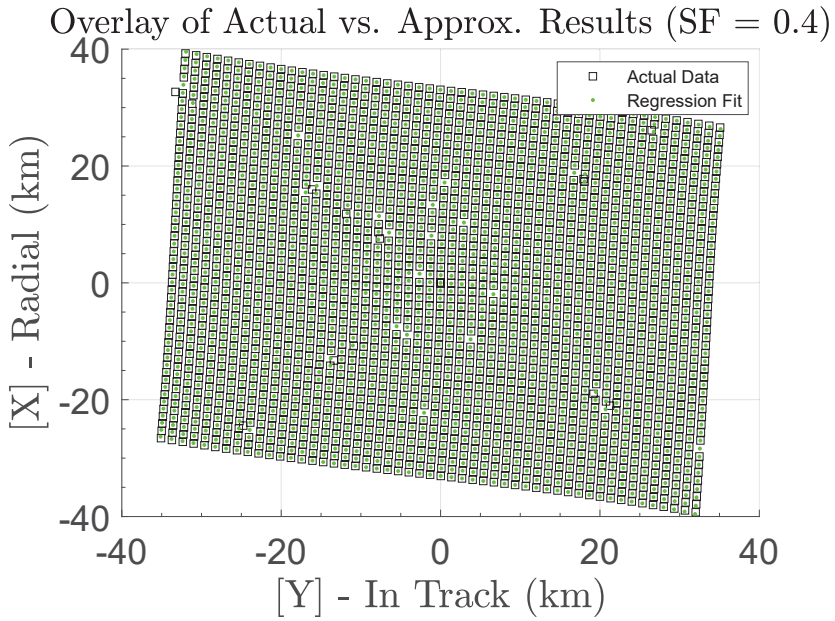


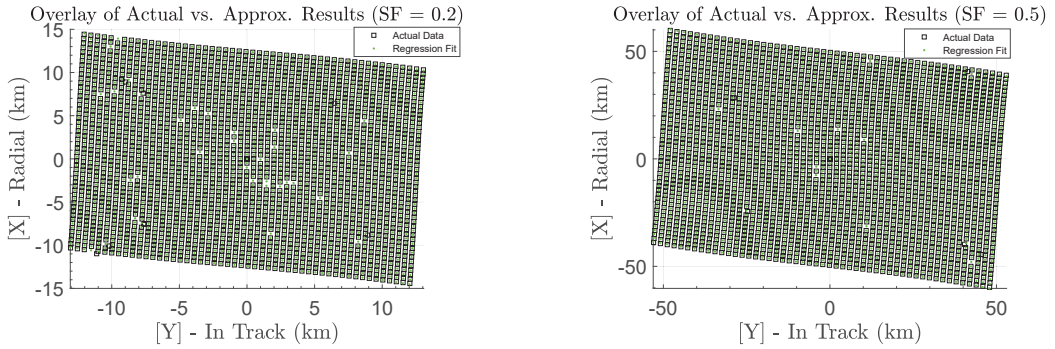
Figure 30. Regression Curve Fit for Capture Positions (SF = 0.4)

The accuracy of the linear regression model is related to the density of the points used to create it. This is considered one of the downsides to the utilization of a linear regression model; a significant computational effort must be taken upfront in order to create the model. For each evader performance scale factor "layer," shown in Figure 30, 2601 points were used to produce fits that have r^2 values given in Table 8.

Table 8. Goodness of Regression Fits for Linear Models

Scale Factor	r_x^2	r_y^2	Points Used
0.2	0.99858	0.99844	2601
0.4	0.99940	0.99895	2601
0.5	0.99992	0.99948	2601
0.7	0.99972	0.99942	2601

The high degree of accuracy in Table 8 across various evader performance scale factors indicates that enough points were chosen for accurate results. Examining the solution sets in Figure 31, shows that the general shape of the solution set remains the same regardless of choice in evader performance scale factor. Between two evader performance scale factors, the only difference is the scaling of the axes values. Thus, it is no surprise that the same level of accuracy can be achieved with the same number of points for each layer.



(a) Capture Positions for SF = 0.2 **(b) Capture Positions for SF = 0.5**

Figure 31. Solution Set Comparison

Table 9 shows an example comparison between the PSO solution (rerun multiple times to ensure its convergence) and the estimated solution from the linear model. The evader performance scale factor chosen for Table 9 matches a "layer" within the linear regression model, meaning that no regression between evader performance

scale factors is required and only one set of functions needs to be evaluated. Table 10 shows the same example comparison, but at an evader performance scale factor that requires the linear regression model to regress between scale factor data. In either case, the results are accurate enough that the linear model can serve as a replacement for the PSO algorithm.

Table 9. Results Requiring No Regression Between Models (SF = 0.50, X_0 = 18 km, Y_0 = 30 km)

Solution	x_f (km)	y_f (km)	t_f (sec)
PSO	-19.109	-27.0483	1961.817
Estimate	-19.107	-27.0571	1961.804
% Difference	0.0085	0.033	0.00066

Table 10. Results Requiring Regression Between Models (SF = 0.43, X_0 = 18 km, Y_0 = 30 km)

Solution	x_f (km)	y_f (km)	t_f (sec)
PSO	-14.357	-20.550	1841.152
Estimate	-14.356	-20.555	1841.141
% Difference	0.0075	0.0028	0.00063

Table 11. Comparison of Computational Speed Between Models Using the Same Computer

Solution Method	Computation Time (sec)
IHM-PSO Method (Non-Parallelized)	1 min 45.55 sec
IHM-PSO Method (Parallelized)	22.86 sec
2D Regression Model	2.17 sec

4.2 Problem B: 3D Linear Regression Model

4.2.1 3D Linear Regression Model.

Results of the 3D linear regression model show similar accuracy to that of the 2D case. On average the 3D linear regression model is able to predict capture positions and time to within 2% of the IHM-PSO data. The source of inaccuracy is largely due to interpolation error and lower density of points used (compared to the 2D case). Recall, in the 2D case there was no interpolation used; linear regression of the evaluated x and y points with respect to evader performance scale factor was performed. This minimized the error by fitting a line to a set of 7+ points. In the 3D case, this methodology is plausible, but too slow. Recall, as mentioned in Section 3.2.1, a single 3D linear regression model is made up of a collection of 2D linear regression models that map (y_i, x_i) to (y_f, x_f) for a given z_0, scale factor, and ψ_0. Thus fitting a linear regression to the evaluated (y_f, x_f, z_f) points like in the 2D case is actually fitting a surface of (y_f, x_f, z_f) points that vary with respect to scale factor and ψ_0. The total number of linear regression models to evaluate becomes substantial, making the total evaluation time akin to the IHM-PSO method. Therefore bilinear interpolation is used instead, which interpolates between scale factor and ψ_0 points, requiring only four 3D linear regression models to be evaluated. In Chang's formulation of a trend surface [56], more 3D models could be used to increase the accuracy, however experimentation showed that adding more 3D models had diminishing return on accuracy and steady impact on computational time. The use of four 3D models balanced both speed and accuracy, resulting in a 3 second computation on average, which is once again substantially faster than the IHM-PSO method. A summary of the computational speed of the 3D linear regression model, baselined against the IHM-PSO method, is shown in Table 12.

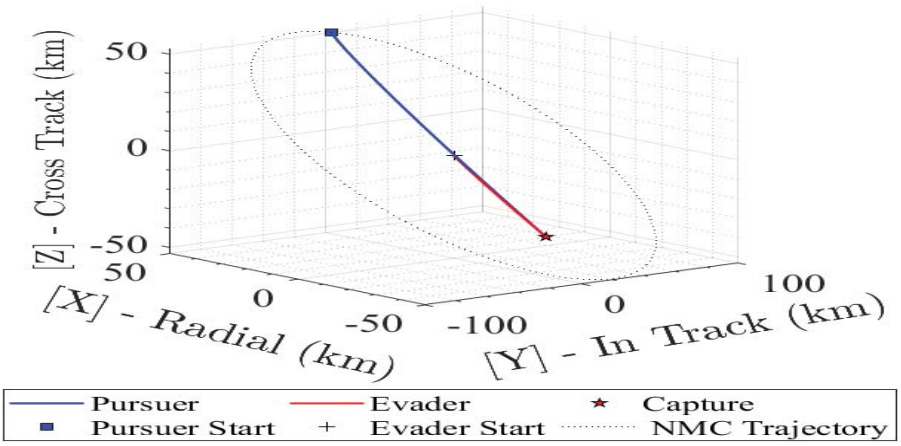


Figure 32. Example of IHM-PSO method

Table 12. Comparison of Computational Speed Between Models

Solution Method	Computation Time (sec)
IHM-PSO Method (Non-Parallelized)	1 min 45.55 sec
3D Regression Model (Surface Regression)	56.10 sec
IHM-PSO Method (Parallelized)	22.86 sec
3D Regression Model (1D Interpolation)	10.06 sec
3D Regression Model (Bilinear Interpolation)	6.03 sec
3D Regression Model (No Interpolation Needed)	1.91 sec

Examining the accuracy of the 3D linear regression model shows that accuracy depends upon the mode of operation. When no interpolation between linear regression models is needed the accuracy is quite high, as shown in Table 13. When interpolation in one direction is required the 3D linear regression model's accuracy worsens slightly, as shown in Table 14 and in Table 15. Finally when both the initial cross-track angle and evader performance scale factor require interpolation, bilinear interpolation is used. This has the largest impact on error from interpolation, especially in regions where either the distance between the four points is large or the range of values of evaluated points is large, as shown in Table 16.

Table 13. Results Requiring No Interpolation (SF = 0.60, X_0 = 40 km, Y_0 = 35 km, Z_0 = 34 km, $\psi_0 = 50°$)

Solution	x_f (km)	y_f (km)	z_f (km)	t_f (sec)
PSO :	-63.494	-38.235	-53.785	2971.423
Estimate :	-63.494	-38.236	-53.802	3005.510
% Diff:	-0.0002	-0.003	-0.03	1.14

Table 14. Results Requiring SF Interpolation (SF = 0.65, X_0 = 43 km, Y_0 = 32 km, Z_0 = 30 km, $\psi_0 = 5°$)

Solution	x_f (km)	y_f (km)	z_f (km)	t_f (sec)
PSO :	-85.853	-34.620	-139.969	3865.096
Estimate :	-85.425	-34.676	-139.131	3873.227
% Diff:	-0.50	-0.16	-0.60	0.21

Table 15. Results Requiring ψ_0 Interpolation (SF = 0.40, X_0 = 50 km, Y_0 = 4 km, Z_0 = 50 km, $\psi_0 = 70°$)

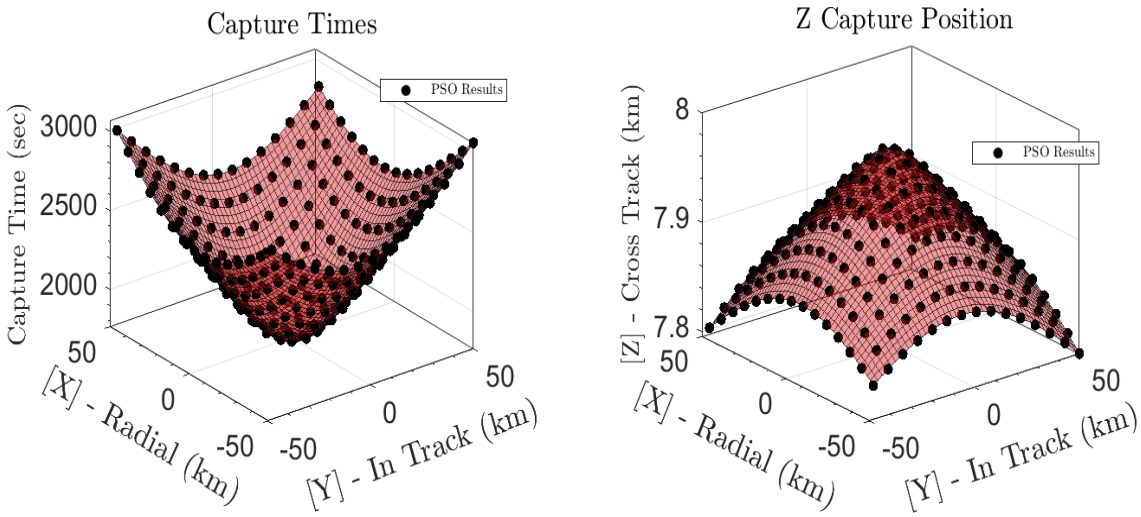
Solution	x_f (km)	y_f (km)	z_f (km)	t_f (sec)
PSO :	-33.729	3.792	-33.536	2629.971
Estimate :	-33.727	3.780	-33.324	2624.318
% Diff:	-0.01	0.30	-0.63	0.22

Table 16. Results Requiring SF and ψ_0 Interpolation (SF = 0.45, X_0 = 10 km, Y_0 = 35 km, Z_0 = 3 km, $\psi_0 = 140°$)

Solution	x_f (km)	y_f (km)	z_f (km)	t_f (sec)
PSO :	-9.197	-27.135	-2.209	1927.696
Estimation :	-9.379	-27.624	-2.170	1898.678
% Diff:	-1.95	-1.79	-1.79	1.52

Looking further inside the 3D linear regression model, each cross-track capture position was estimated using a cubic interpolation of final cross-track position (z_f) plotted with respect to initial radial (x_i) and in track (y_i) positions. Capture time (t_f) was also found in this manner. Recall, as discussed in Section 3.1.3, linear regression

could not be used in cases where the funnel shape constricted too narrowly for a polynomial surface to adequately approximate. It was found that the funnel exists only for z heights that approach zero, and the larger the magnitude of z, the more parabolic the z capture surface becomes. Given that some z surfaces still exhibited the funnel behavior, interpolation was chosen over regression for both the capture time and z height surfaces. The results of cubic interpolation can be seen in Figure 33.



(a) Capture Time (sec) **(b) Z Capture Position (km)**

Figure 33. Cubic Interpolation Results

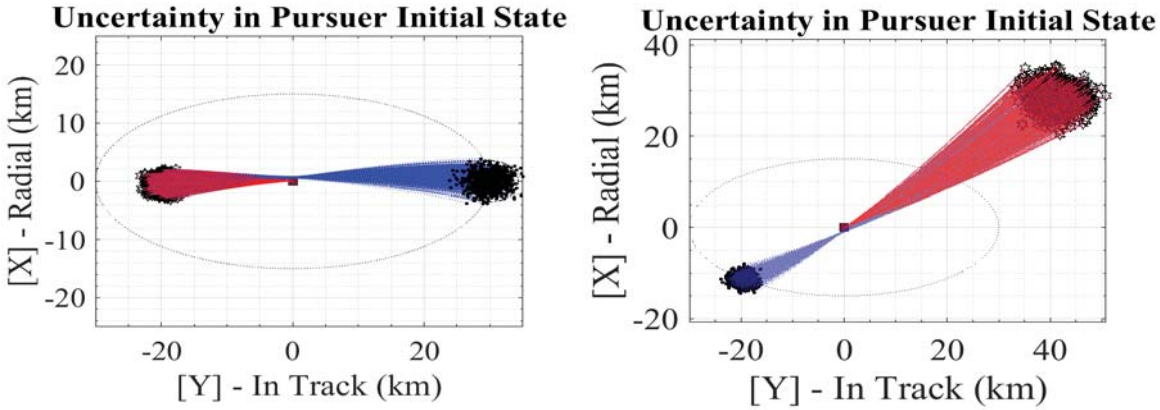
In summary, the 3D linear regression model sufficiently predicts the capture position and time across the solution space of all relative orbits within 50 km of the evader. A small accuracy penalty of approximately $1-2\%$ error was paid for 9.3 times faster computational speed by using bilinear interpolation over calculation of a trend surface. The speed of the 3D linear regression model is quick enough for possible onboard computation. Furthermore it only takes up 55 MB of memory, which is impressive considering the raw data used to make it is approximately 3 GB of data.

4.3 Problem C: Parameter Study of 2D/3D Game

4.3.1 2D Analysis.

Running the PSO algorithm, developed by Prince et al. [27] and modified to work as a function, produced results that form the data set for the linear regression model. An example of the output from the iterated routine is shown in Figure 34.

As an observation from the shown results, the family of blue pursuer trajectories share a point of inflection near the center of the NMC initial trajectory. This phenomenon persists throughout the range of evader performance scale factors, sizes of covariance ellipses, and initial phasing angles.



(a) PSO Output for SF = 0.4 **(b) PSO Output for SF = 0.7**

Figure 34. Solution Set Comparison

The inflection of the capture positions are shown in Figure 35, where the results are colorized based upon the sign of the in-track component of the final capture position. The colorization shows how pursuer starting positions and the final capture positions are opposite one another, inflected through the "pinch point" that occurs near the center of the initial NMC trajectory.

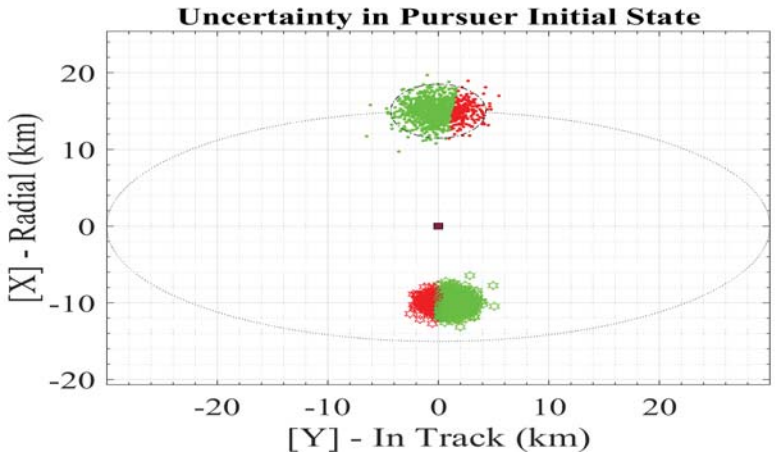


Figure 35. Colorized PSO Output (SF = 0.4)

Figure 36 showcases a variety of different results. First, the green pursuer starting regions are phased roughly 180 degrees from the red capture regions (i.e. they are on the other side of the NMC). The phasing of the capture region in relation to the initial starting region is not affected by the choice of the NMC's semi-major axis or by the evader performance scale factors. However, changing the evader performance scale factor affects both the distance of the capture region from the evader starting position and the size of the covariance ellipse, as shown in Figure 37.

It can be seen in Figure 37 that selecting an evader performance scale factor greater than 0.5 (corresponding to less penalization on the evader's performance) results in final capture regions that are outside the pursuer's initial NMC trajectory. As the evader becomes as capable as the pursuer, the terminal time (as well as the distance covered) increases. Likewise, when the evader performance scale factor is less than 0.5 (the evader's performance is heavily handicapped), then the pursuer easily catches the evader inside of the initial NMC trajectory. Based on this analysis, a good rule of thumb is if the pursuer is at least twice as fast as the evader (i.e. the evader performance scale factor < 0.5) then the center of the capture regions, or average capture position, will always occur inside the area bounded by the pursuer's initial instantaneous NMC trajectory.

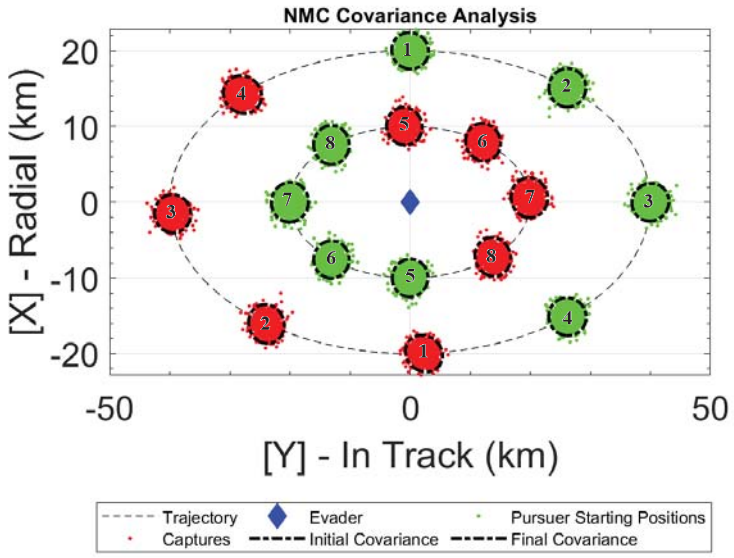
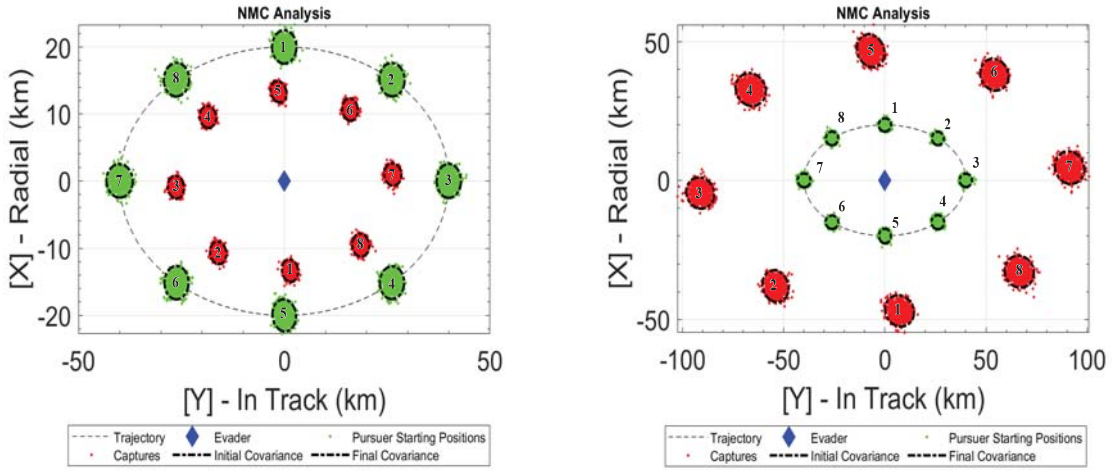


Figure 36. Covariance Effects Based on Angle and Range (SF = 0.5)

Figure 36 shows that the covariance ellipses for the starting regions are approximately the same size as the capture regions, which is confirmed by Table 17. However, this is only the case for a evader performance scale factor of 0.5. When the scale factor is anything else, the size of the covariance ellipse changes depending upon the performance scale factor chosen, as seen in Figure 37 for an scale factor of 0.4 and 0.7. Regarding the size of the covariance ellipses, the scale factor of 0.5 again seems to be the division upon which the size of the covariance ellipses start to either contract or expand. For example, depicted in Figure 36 and given in Table 17, the percent change from initial to final variances are on the order of one percent. When the scale factor is raised from 0.5 to 0.7, the percent changes in variance are on the order of 450%, or approximately 5.5 times their initial values as seen in Table 18.

Figure 38 shows how the orientation of the covariance ellipses is mostly preserved as a result of initial variance and angle. Plotted are two groups of runs at the same range, and for the same evader performance scale factor. In each run, the choice of initial variances in pursuer position is switched. The results indicate that the



(a) Covariance Effects Based on Angle and Range (SF = 0.4)

(b) Covariance Effects Based on Angle and Range (SF = 0.7)

Figure 37. Covariance Effects Based on Angle and SF

final covariance ellipse for the terminal capture positions is independent of phasing angle. The results also show that the orientation of the covariance ellipses is largely preserved. There is a slight tilt of the capture position covariance ellipses that seems to indicate that the capture position variances are slightly affected as a result of the solution process, which is worth looking into further.

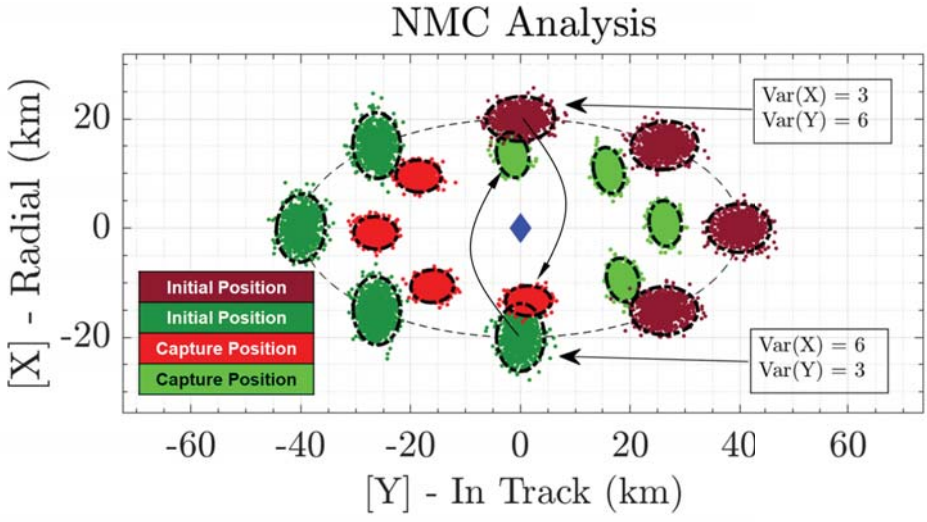


Figure 38. Covariance as a Function of Angle and Variance

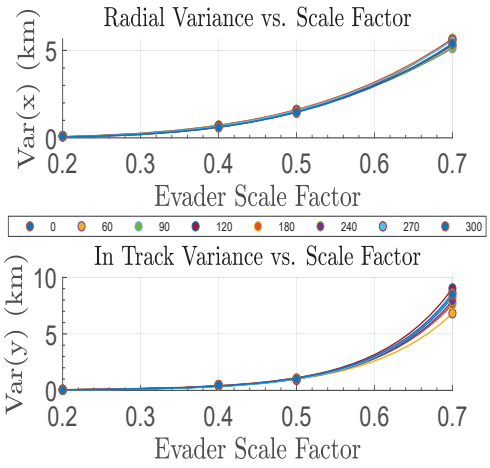
Table 17. Variance as a Function of Relative Phasing Angle, ν_r (SF = 0.5, Initial Var(x) = 1.5 km, Initial Var(y) = 1 km)

ν_r	$\text{Var}_0(x)$	$\text{Var}_f(x)$	% Change	$\text{Var}_0(y)$	$\text{Var}_f(y)$	% Change
0	1.583	1.592	0.577	1.037	1.075	3.649
60	1.531	1.554	1.501	0.941	0.914	-2.793
90	1.570	1.569	-0.083	0.954	0.970	1.749
120	1.495	1.495	-0.027	1.066	1.091	2.309
180	1.430	1.442	0.837	0.973	0.987	1.365
240	1.562	1.590	1.816	0.945	0.904	-4.309
270	1.510	1.512	0.138	0.992	0.996	0.470
300	1.496	1.484	-0.786	1.042	1.106	6.139

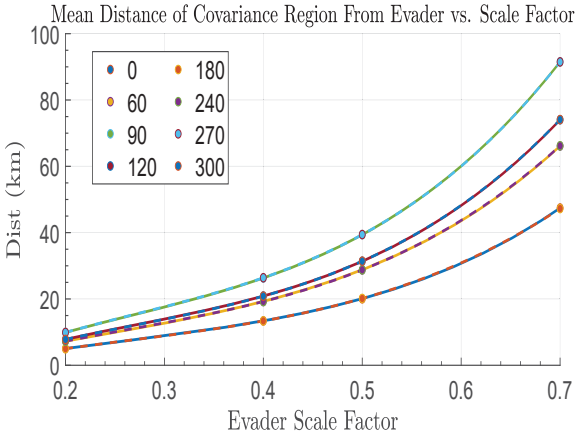
Table 18. Variance as a Function of Relative Phasing Angle, ν_r (SF = 0.7, Initial Var(x) = 1.5 km, Initial Var(y) = 1 km)

ν_r	$\text{Var}_0(x)$	$\text{Var}_f(x)$	% Change	$\text{Var}_0(y)$	$\text{Var}_f(y)$	% Change
0	1.367	7.567	453.414	1.041	5.955	471.946
60	1.368	7.661	460.089	0.993	5.154	418.831
90	1.479	8.136	450.280	1.037	5.661	445.900
120	1.411	7.661	442.841	0.950	5.680	498.424
180	1.424	7.900	454.848	0.968	5.463	464.188
240	1.444	8.142	463.834	1.039	5.233	403.733
270	1.548	8.571	453.686	1.046	5.583	433.957
300	1.481	8.087	445.975	1.056	6.127	480.506

A closer look at how the variance (in both the radial and in-track capture positions) varies as a function of evader performance scale factor is shown in Figure 39a. The results show that the variance in the capture region increases as a function of evader performance scale factor, reaffirming what was already seen in the previous plots where a larger scale factor resulted in a larger covariance ellipse. Among different phasing angles for each evader performance scale factor, the variances show good agreement, however they start to diverge from one another at large scale factors. Finally, the variance in the in-track component of the capture positions increase much more quickly than the variance in the radial component. In fact, the variance in the radial direction best fits a power curve ($f(x) = ax^b$) whereas the in-track direction best fits an exponential curve ($g(x) = ae^{bx}$), both with $b > 0$.



(a) Variance as a Function of SF



(b) Movement of Capture Region as a Function of Evader Scale Factor

Figure 39. Covariance Results

Distances from the evader to the center of each capture region, (as a function of evader performance scale factor), is shown in Figure 39b. As the scale factor increases (as the evader is less penalized in terms of performance) the average distance the evader moves increases, which makes intuitive sense. In Figure 39b, there are a total of eight different angles plotted, however each angle has paired off with the angle 180 degrees across from it. Interestingly, there are four total groups instead of three because of the slight positional shift in the centers of the capture region best seen in Figure 37a. This positional shift disrupts the would-have-been symmetry of capture regions which results in the distances of the pairings {2, 6} and {4, 8} (from Figure 37a) not having equivalent distances from the evader.

4.3.2 3D Analysis.

Results from the 3D analysis compliment the 2D analysis, and verify that the trends that exist in the 2D planar case also exist in the more general 3D case. Shown

in Figure 40 is an example of the 3D analysis showing how the capture covariance ellipsoid changes size and position based upon evader performance scale factor.

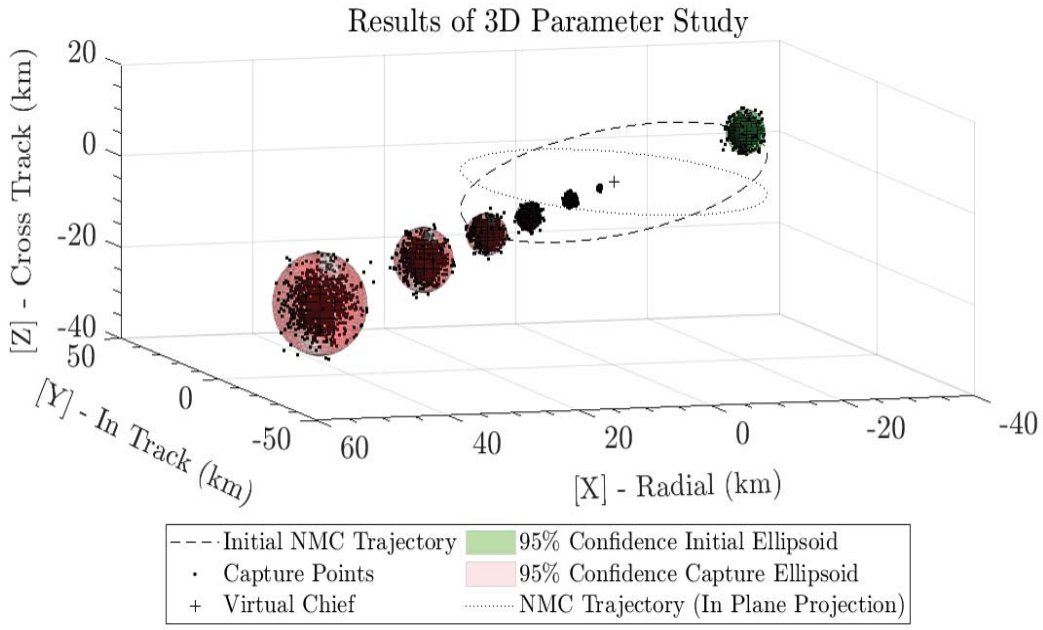


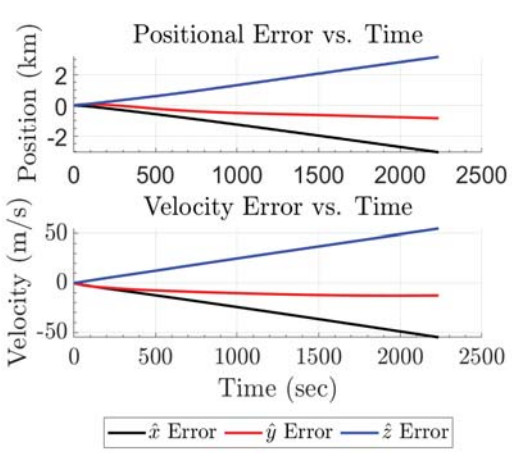
Figure 40. Movement and Size of Capture Region as a Function of Evader Scale Factor

4.4 Problem D: Validation of PSO-IHM Solutions

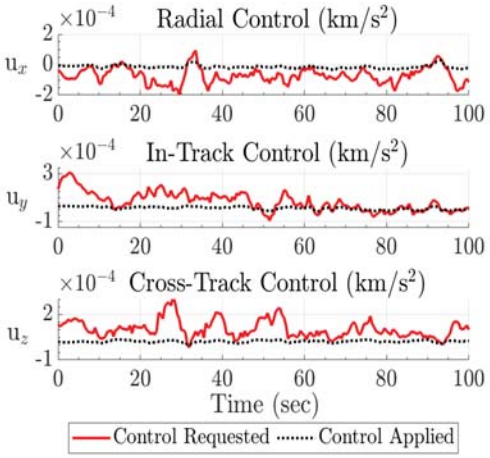
The trends shown in Section 4.3 have been shown for a 2D and 3D analysis, however they both assume that the states are propagated using linearized dynamics. Furthermore it is assumed that the flight trajectories generated by the IHM-PSO method are able to be flown by the satellites. This section investigates both the accuracy of the linearized assumptions made in this work and validates that the generated trajectories are flyable by running the open loop trajectories through a Guidance and Navigation framework [45] in a closed-loop manner. Both the accuracy of the orbits and ability to be flown are crucial to this research being applicable to real-world mission planning.

4.4.1 Flying the IHM-PSO Generated Orbits.

The GNC system [45] was used to determine if the IHM-PSO trajectories are able to be flown in a closed-loop manner. The results appear to indicate that the success in "flying" the trajectory depends upon the distance traveled by the spacecraft. Over relatively short distances the LQR controller is able to satisfactorily track the desired trajectories. However as the distance traveled increases, the LQR fails to adequately control the spacecraft motion. This was discovered after reviewing results where the evader trajectory was successfully flown and the pursuer trajectory (which travelled approximately double the distance) suffered from poor accuracy. Investigating this further yields an indication that the problem lies with the available control acceleration. As formulated in the IHM-PSO method, the initial control acceleration, a_0, without any modification from performance scale factors is 3.43×10^{-5} km/sec, which is much lower than the requested acceleration from the LQR controller. While the IHM-PSO method takes into account the control bounds, the GNC system propagates the dynamics with the NERMs instead of the HCw equations. The resulting error over time has the effect of saturating the control applied to the system, which negatively impacts the controller's ability to keep the spacecraft on course. As the distance travelled by the spacecraft increases, so does the time that the control is saturated. The results, shown in Figures 41 and 42, indicate that the LQR controller is able to deal with a limited amount of saturation, but over time the difference between the estimated (actual) trajectory and the reference trajectory steadily grows, driving the requested acceleration larger, which then results in more substantial saturation of the requested input.

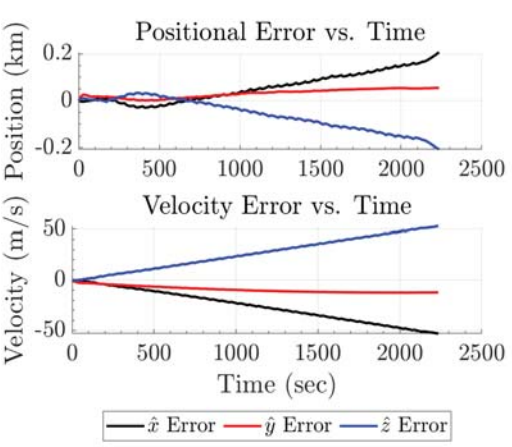


(a) Error in States

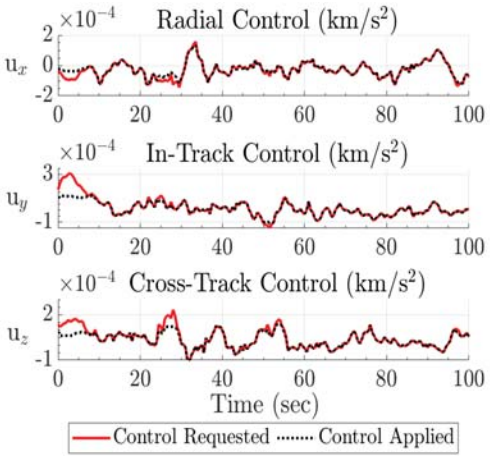


(b) Requested vs. Applied Control

Figure 41. Pursuer Trajectory (SF = 1)



(a) Error in States



(b) Requested vs. Applied Control

Figure 42. Pursuer Trajectory (SF = 4)

The problem of too low initial control accelerations are exacerbated by the application of performance scale factors. As discussed in the Section 3.1.3, the evader performance scale factor was chosen as a value between 0.05 and 0.70 and the pursuer performance scale factor was left as 1.0. In the case of the evader, dividing the already too low initial control acceleration by as much as a factor of ten resulted in even less ability of the evader to maintain their course in the closed-loop simulation. Note that in the open loop formulation, dividing the evader performance scale factor by as much

as ten was plausible because it was not being evaluated whether or not that amount of reduction was realistic in terms of "flying" the trajectory that was solved. Mathematically the trajectories are solvable for any positive value of initial acceleration. Realistically, however, picking too low of an initial control acceleration term results in a "non-flyable" trajectory. Figure 43 shows that using an evader scale factor of 1.0 and a pursuer performance scale factor of 2.0 resulted in adequate closed-loop performance, and thus a likely more realistic value for a_0. It should be pointed out that this simulation assumes a single electric propulsion thruster, which is not likely the case for real systems, which would have cold gas systems for maneuvering as well. Thus as demonstrated by the results of the GNC closed-loop simulation, the initial control acceleration chosen in this research, based off values chosen by Prince [27], appears slightly optimistic for adequate performance over distances, but is plausible for short distances where LQR can compensate with a saturated control.

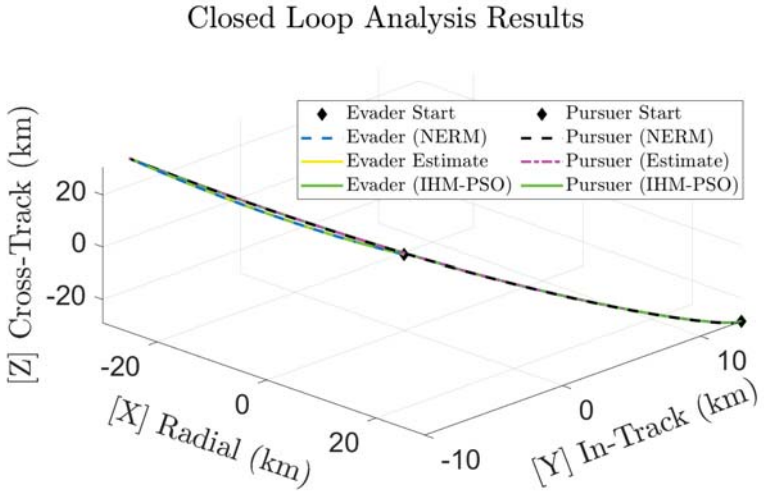


Figure 43. Flown IHM-PSO Trajectories

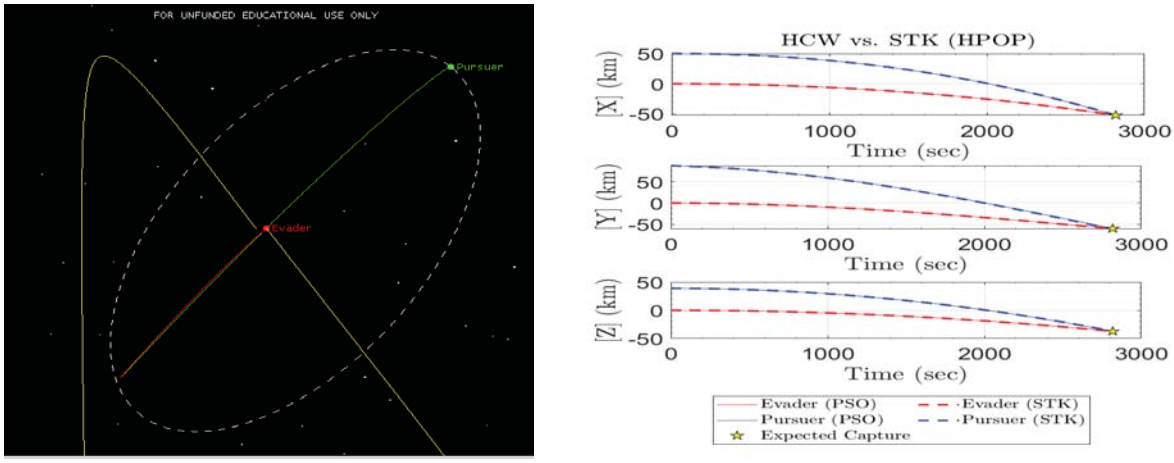
It is important to point out that the GNC system is designed to track the particular relative orbit, and not to pursue/evade the other player. This distinction is important because the measurements made by the sensors establish position/velocity that is used

to drive the error between the reference trajectory and the actual trajectory to zero. In an actual pursuit-evasion game the pursuer and evader would take measurements of the position of the other player and then use these measurements to make course corrections to minimize/maximize the capture time (depending upon the player). As such, because of the GNC's primary purpose as a trajectory tracker, there are miss distances between the pursuer and evader, which result from things such as saturated control, propagation error resulting from the difference in NERMs vs HCW equations, etc. It was not the intent of this section to provide a solution to the pursuit-evasion game in a closed-loop manner, but instead to validate that the open loop trajectories can be adequately flown. The next step in the evolution of this research would be to tie in measurements made on the other player to enable a course correction pursuit-evasion game. Thus it has been shown that the GNC framework provides evidence that these trajectories are "flyable" with a proper choice of initial control acceleration.

4.4.2 Accuracy of IHM-PSO Generated Orbits.

To gauge if the IHM-PSO solutions are accurate representations of what on orbit conditions would be like, the trajectories are planned with Astrogator and propagated with HPOP, both of which are tools within AGI's STK software. A selected solution from the IHM-PSO algorithm was purposely chosen at the 50 km range such that the game covers the maximum distance the linear regression model is capable of working with. As a result, the error that accumulates from the linearization assumption will be the maximum expected error, given the distance traveled for all games studied is the furthest in this selected solution. An evader scale factor of 0.50 was chosen to validate the trends analyzed in Section 4.3; namely that the expected capture position is 180° in phase, and on the initial relative NMC orbit. The results of the numerically propagated trajectories are shown graphically within STK in Figure 44a, and plotted

with respect to time in Figure 44b for the long range game. The video file for this STK simulation is attached to this pdf under the paperclip icon in Adobe.



(a) STK Visualization **(b) STK Propagated Trajectories**

Figure 44. STK Propagation Results

In Figure 44a, the 3D pursuit-evasion game with high fidelity propagation shows a variety of results. First, the validity of using linearized HCW dynamics is proven by STK results matching those found with the IHM-PSO method. The miss distance as a result of higher-fidelity propagation is 434.2 meters, which is largely insignificant compared to the total amount of distance traveled by each spacecraft. This is most likely because the duration of all games studied satisfy the requirement of HCW dynamics in that the total time studied is much less than the orbital period. Second, the trends and "rules of thumb" discussed in Section 4.3 are shown to be valid for not only HCW dynamics, but valid in general for higher-order dynamics as well. This is significant because it shows that the trends shown are not only applicable in an idealized world of linearized dynamics, but can see application in real-world mission operations.

Combined together, the GNC framework and the STK propagation support the claim that this research is applicable to real-world scenarios and can be used for future

space mission planning. The linear regression model is capable of quickly determining the estimated capture time and position for a generalized pursuit-evasion scenario that has been shown to work quite well even in environments that contain perturbation and nonlinearities that are not modeled in the dynamics formulation underpinning the linear regression model data set. With a nonlinear set of dynamics, with the inclusion of perturbative forces, a new data set could be constructed to yield an even more accurate linear regression model.

4.4.3 Chapter Summary.

This chapter has shown the results for each problem studied in this thesis. Problem A documented that the results of the 2D linear regression model achieved an accuracy within 2% of the optimal control solutions and could be solved for approximately 3.8 times faster that the IHM-PSO method. Problem B extended the linear regression model to include cross-track motion, and saw similar performance to that of the 2D case. To retain the speed advantage over the optimal control solution method, a small accuracy penalty of $1-2\%$ was paid. Thus, by choosing a local approximation technique over a global one, the 3D regression technique increased its computational speed by a factor of 9.3. Problem C took the developed linear models and created a large scale parameter study that uncovered numerous trends for a generalized pursuit-evasion game. These trends were presented as the beginnings of TTP development for this type of pursuit-evasion game. Finally, Problem D showed that the pursuit-evasion trajectories solved by the IHM-PSO method are both accurate and largely viable for use in real-world environments with a sufficient engine setup. In addition, Problem D showed that the trends identified in Problem C remained valid when higher fidelity dynamics were used to propagate the states, reaffirming they're validity for real-world mission planning.

V. Conclusion

This chapter provides a conclusion to the thesis by summarizing important points to each part of the thesis, providing ideas for future work, and identifying specific contributions to the DoD.

A linear regression model has been shown to effectively map the initial pursuer relative starting positions to the terminal capture positions in a zero-sum pursuit-evasion differential game, effectively answering the first research question. These linear regression models are useful in situations where computational speed and efficiency are prioritized, and where the control profiles are not required. The linear regression model is simply a group of functions, and as such, could easily be implemented on-board a spacecraft for quick evaluations and possible autonomous mission planning.

Answering the second research question, this research has shown that the linear regression model is just as accurate as the PSO algorithm that was used to create it. While the results are within 2% of the IHM-PSO algorithm, they provide consistent results whereas the IHM-PSO returns optimal results only $70 - 90\%$ of the time [6]. Thus it has been shown that the linear regression models are both accurate and reliable enough to be utilized in real-world mission planning and TTP development. In certain cases, like the scenarios studied in this research effort, linear regression models could replace optimal control solvers in determining the optimal control solution.

The third research question sought to investigate large scale parameter studies aimed at finding trends for a generalized pursuit-evasion game. This research has shown how a parameter study that would have taken a month to compute with the IHM-PSO method instead took only 15 minute to perform. Born from the results of this parameter study, several trends were identified. First, regardless of the range, the capture positions are approximately 180 degrees in phase from the initial starting

positions. Changing the evader performance scale factor affects both the distance of the capture regions from the evader's starting position and the size of the capture position's covariance ellipse. Finally, in all the scenarios examined, the trajectories shared a point of inflection close to the LVLH origin. This represents a point at which all trajectories will pass through at some point in time, marking a prime place for an interceptor spacecraft to be placed in the "Lady, Bandit, Bodyguard" differential game [32].

The fourth and fifth research questions aimed at determining the viability of the generated IHM-PSO solutions. In particular the solution trajectories were used as guidance in a trajectory tracking GNC system, and propagated with high fidelity simulation software. This research has shown that with the use of a developed GNC closed-loop control system, along with STK's high fidelity propagation tools, have provided evidence to support the claim that the linear regression model can likely be used in real-world applications. Specifically, the GNC system showed that the trajectories are flyable given the right setup of controller and filtering algorithms. The STK propagation showed that the error due to assuming linearized dynamics is sufficiently small over the quick time durations studied in this research. A higher fidelity propagation also validated the "rules of thumb" that are proffered as a means for characterizing how pursuit-evasion games of this type will be played out. Doing so allows the linear regression model and associated trends to better inform mission planners for real-world operations.

In summary, by using a several gigabytes of data, it is possible to create a linear regression model that maps the pursuer initial positions to final capture positions. The linear regression model boasts the advantages of being fast, requires minimal computational power, and takes up only 55 MB of memory. The original data set was created by transforming the optimal control problem into a parameter optimization

problem through the use of an indirect heuristic method. The resulting parameter optimization problem was then solved with a particle swarm optimization routine. Applying the linear regression model to the collected data proved useful in conducting a parameter study of an imperfect information pursuit-evasion differential game. Doing so showcases the power of the linear regression model to quickly and efficiently produce data useful in identifying trends to help future mission planning for space-based proximity operations.

5.1 Future Work

There are a number of ways that this research could be used in the future. By implementing a more complex (nonlinear) set of dynamics into the IHM-PSO algorithm, two improvements could be made. First, the accuracy of the orbital results would improve, lending to better open loop guidance profiles for closed-loop trajectory planning. Adding in environmental disturbances, such as J_2 gravitational effects could bring the results closer to what one would expect on orbit. Second, the evader performance scale factors above 0.7 could be examined since there would no longer be linearization distances and times to keep small. This would allow games with satellites of almost equal performance to be analyzed, where the distances travelled could potentially be great.

Along this same vein, by reformulating the relative orbital elements to solve for the required initial velocities when using TH dynamics would allow games of arbitrary eccentricity to be studied. A study analyzing how accurate the linear regression models developed in this research are when compared to slightly eccentric games could provide further credibility regarding the real-world use of the developed linear regression models, while at also expanding the linear regression model's flexibility to handle elliptic reference orbits.

This research has produced a large amount of input-output data and optimal control trajectories. A neural network could be given this research's collected PSO data to train on in order to predict the trajectories for arbitrary cases. Doing so could result in a model that retains the advantages of being fast (similar to the linear regression model), but could also provide the entire trajectory profile, thereby sacrificing none of the results (such as the state information) between the initial and terminal positions. Conversely, the underlying manifolds shown with the linear model could be used to validate the optimality of results obtained by other AI developed solutions.

5.2 Contributions and Impact to DoD

This research has pushed the boundaries on how deterministic games can be iteratively solved for in order to model stochastic effects. The work regarding the 2D linear regression model has been published in a SciTech 2020 paper [43], and the GNC framework published in a conference paper at the 2020 Institute of Navigation (ION) Position Location and Navigation Symposium (PLANS) [45]. This research effort was also presented at the 2020 Dayton-Cincinnati Aerospace Sciences Symposium in an effort to share and collaborate with other researchers. The linear regression models have thus been proven to work mathematically and in highly realistic simulated space environments. These linear regression models provide the Department of Defense with new tools to help develop TTPs to quickly characterize and respond to pursuit-evasion scenarios. By providing intuitive "rules of thumb", this research enables future mission planners to be able to effectively gauge terminal capture regions with simple visual aids, without running optimal control problems. By evaluating the linear regression model, future mission planners can gain quick estimates for duration of the "what-if" pursuit-evasion scenario as well as distance travelled. These metrics

can be used to determine time evading assets are taken off mission as well as total fuel consumption required (based off first order time and distance). In summary, this research provides methods intended to help develop DoD develop TTPs to enhance capabilities in a "congested, contested, and competitive" space environment.

Bibliography

[1] R. Bate, D. Mueller, and J. White, "The fundamentals of astrodynamics," Dover Publications, pp. 13–14, 1971.

[2] K. Alfriend, S. Vadali, P. Gurfil, J. How, and L. Breger, "Spacecraft formation flying," Dover Publications, 2010.

[3] B. A. Conway, "Spacecraft trajectory optimization," Cambridge University, pp. 1–2, 37–40, 45–46, 2010.

[4] D. L. Kunz, "Intermediate dynamics for aeronautics and astronautics," Headmaster Press, pp. 35–36, 2015.

[5] A. Jagat, "Spacecraft relative motion applications to pursuit-evasion games and control using angles-only navigation," Auburn University, May 2015.

[6] E. Prince, "Optimal finite thrust guidance methods for constrained satellite proximity operations inspection maneuvers," Air Force Institute of Technology, September 2018.

[7] W. H. Clohessy and R. S. Wiltshire, "Terminal Guidance System for Satellite Rendezvous," J. Aerosp. Sci., vol. 27, no. 9, pp. 653–658, 1960.

[8] C. George, "Optimal and robust neural network controllers for proximal spacecraft maneuvers," Air Force Institute of Technology, March 2019.

[9] E. W. Weisstein, "Mean Motion." `http://scienceworld.wolfram.com/physics/MeanMotion.html`, 2007.

[10] J. Sullivan, "Comprehensive survey and assessment of spacecraft relative motion dynamics models," Journal of Guidance, vol. 40, no. 8, p. 1851, 2017.

[11] D.-W. Gim and K. T. Alfriend, "State Transition Matrix of Relative Motion for the Perturbed Noncircular Reference Orbit," Journal of Guidance, Control, and Dynamics, vol. 26, no. 6, pp. 956–971, 2003.

[12] G. E. De Bruijn, F. and J. How, "Comparative Analysis of Cartesian and Curvilinear Clohessy–Wiltshire Equations," Journal of Aerospace Engineering, Sciences and Applications, vol. 3, no. 2, p. 1–15, 2011.

[13] 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.

[14] T. Lovell and D. Spencer, "Relative orbital elements formulation based upon the clohessy-wiltshire equations," The Journal of the Astronautical Sciences, vol. 61, 02 2015.

[15] T. Lovell and S. Tragesser, "Guidance for relative motion of low earth orbit spacecraft based on relative orbit elements," AIAA/ASS Astrodynamics Specialist Conference and Exhibit, 08 2004.

[16] M. Kelly, "An Introduction to Trajectory Optimization," Society for Industrial and Applied Mathematics, vol. 26, no. 59, p. 849–904, 2017.

[17] D. E. Kirk, "Optimal Control Theory: An Introduction," Dover Publications, pp. 17–34, 233, 1998.

[18] A. E. Bryson, Dynamic Optimization. Addison-Wesley, 1999.

[19] J. Stupik and B. Conway, "Optimal pursuit/evasion spacecraft trajectories in the hill reference frame," AIAA Astrodynamics Specialist Conference, 2012.

[20] V. M. Guibout and D. J. Scheeres, Modern Astrodynamics. Elsevier, 1 ed., 2006.

[21] L. Biegler and V. Zavala, "Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide optimization," Computers & Chemical Engineering, vol. 33, no. 3, pp. 575–582, 2009.

[22] P. Gill, W. Murray, and M. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," SIAM Journal on Optimization, vol. 12, pp. 979–1006, 04 2002.

[23] 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 Trans. Math. Softw, vol. 41, no. 1, p. 37, 2014.

[24] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948, 1995.

[25] M. Pontani and B. Conway, "Optimal low-thrust orbital maneuvers via indirect swarming method," Journal of Optimization Theory and Applications, vol. 162, no. 1, pp. 272–292, 2014.

[26] R. Isaacs, Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization. Mineola, New York: Dover Publications, Inc., 1965.

[27] E. R. Prince, J. A. Hess, R. G. Cobb, and R. W. Carr, "Elliptical Orbit Proximity Operations Differential Games," Journal of Guidance, Control, and Dynamics, vol. 42, no. 7, pp. 1458–1472, 2019.

[28] D. Shen, K. Pham, E. Blasch, H. Chen, and G. Chen, "Pursuit-evasion orbital game for satellite interception and collision avoidance," SPIE Defense, Security, and Sensing Conference, 2011.

[29] 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.

[30] K. Horie and B. Conway, "Optimal Fighter Pursuit-Evasion Maneuvers Found via Two-Sided Optimization," Journal of Guidance, Control, and Dynamics, no. 1, pp. 105–112, 2006.

[31] S. W. Yifang Liu, Renfu Li, "Orbital Three-Player Differential Game Using Semi-Direct Collocation with Nonlinear Programming," 2nd International Conference on Control Science and Systems Engineering, pp. 217–222, 2016.

[32] I. Rusnak, "The Lady, the Bandit and the Body-guard game," 2004.

[33] M. P. Ryan W. Carr, Richard G. Cobb and S. Pierce, "Solution of a Pursuit–Evasion Game Using a Near-Optimal Strategy," Journal of Guidance, Control, and Dynamics, vol. 41, no. 4, pp. 842–850, 2018.

[34] T. D. Woodbury and J. E. Hurtado, "Adaptive play via estimation in uncertain nonzero-sum orbital pursuit evasion games," AIAA Sp. Astronaut. Forum Expo., pp. 1–15, 2017.

[35] N. Satak, "Behavior Learning in Differential Games and Reorientation Maneuvers," Texas A&M University, 2013.

[36] K. a. Cavalieri and N. Satak, "Incomplete Information Pursuit-Evasion Games with Uncertain Relative Dynamics," AIAA Guidance, Navigation, and Control Conference, pp. 1–8, 2014.

[37] R. K. Maloy, K. Y. Lee, and L. H. Sibul, "A pursuit-evasion differential game in relative polar coordinates with state estimation," Proceedings of 1995 American Control Conference - ACC'95, vol. 3, pp. 2463–2467 vol.3, June 1995.

[38] F. Amato, M. Mattei, and A. Pironti, "Guaranteeing cost strategies for linear quadratic differential games under uncertain dynamic," 07 1997.

[39] W. Willman, "Formal solutions for a class of stochastic pursuit-evasion games," IEEE Transactions on Automatic Control, vol. 14, pp. 504–509, October 1969.

[40] R. Behn and Yu-Chi Ho, "On a class of linear stochastic differential games," IEEE Transactions on Automatic Control, vol. 13, pp. 227–240, June 1968.

[41] I. Rhodes and D. Luenberger, "Differential games with imperfect state information," IEEE Transactions on Automatic Control, vol. 14, pp. 29–38, February 1969.

[42] A. Antoniades, H. J. Kim, and S. Sastry, "Pursuit-evasion strategies for teams of multiple agents with incomplete information," 42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475), vol. 1, pp. 756–761 Vol.1, Dec 2003.

[43] D. Linville and J. Hess, "Linear regression models applied to spacecraft pursuit evasion differential games," SciTech, 2020.

[44] B. Wie, "Space vehicle dynamics and control," American Institute of Aeronautics and Astronautics, Inc., p. 172, 2008.

[45] W. Harris, D. Linville, H. Joshuah, and R. Cobb, "Development of gnc for optimal relative spacecraft trajectories," Position Location and Navigation Symposium, 2020.

[46] P. S. Maybeck, Stochastic Models, Estimation, and Control. 111 Fifth Avenue, New York: Academic Press, 1979.

[47] K. Ogata, Modern Control Engineering. Prentice Hall, 5 ed., 2010.

[48] S. Milton and J. Arnold, Introduction to Probability and Statistics. McGraw-Hill, 4 ed., 2003.

[49] T. Lacey, "Tutorial: The Kalman Filter," MIT, 1998.

[50] F. Orderud, "Comparison of Kalman Filter Estimation Approaches for State Space Models with Nonlinear Measurements," Scandinavian Conference on Simulation and Modeling, 2005.

[51] S. J. Julier and U. J. K, "New extension of the kalman filter to nonlinear systems," SPIE, vol. 3068, 1997.

[52] S. Milton and J. Arnold, "Introduction to Probability and Statistics," McGraw-Hill Higher Education,, 2003.

[53] Mathworks, "Linear regression." `https://www.mathworks.com/help/matlab/data_analysis/linear-regression.html`, 2019.

[54] P. J. Davis, "Interpolation and Approximation," Blaisdell Publishing Company, p. 25, 1963.

[55] R. Wang, "2-d interpolation." `http://fourier.eng.hmc.edu/e176/lectures/ch7/node7.html`, 2020.

[56] K.-t. Chang, "Introduction to Geographic Information Systems," McGraw-Hill, 2009.

[57] M. Pontani and B. Conway, "Minimum-fuel finite-thrust relative orbit maneuvers via indirect heuristic method," Journal of Guidance, Control, and Dynamics, vol. 38, pp. 913–923, May 2015.

[58] D. F. Spendel, "Parameter Study of an Orbital Debris Defender using Two Team, Three Player Differential Game Theory," Air Force Institute of Technology, 2018.

[59] Mathworks, "isoutlier." https://www.mathworks.com/help/matlab/ref/isoutlier.html, 2019.

[60] Mathworks, "List of library models for curve and surface fitting." https://www.mathworks.com/help/curvefit/list-of-library-models-for-curve-and-surface-fitting.html#btbcvnl, 2019.

[61] A. Johnson, "Error ellipse." https://www.mathworks.com/matlabcentral/fileexchange/4705-error_ellipse, 2015.

[62] AGI, "System tool kit (stk)." https://www.agi.com/home, 2020.

[63] AGI, "High-precision orbit propagator (hpop)." http://help.agi.com/stk/index.htm#hpop/hpop.htm, Jan 2020.

[64] A. S. LeValley and B. C. George, "cont2imp," 2019.

[65] AGI, "Attitude file format (*.a)." http://help.agi.com/stk/index.htm#stk/importfiles-01.htm, Jan 2020.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
03/26/2020	Master's Thesis	Sept 2018 - Mar 2020

4. TITLE AND SUBTITLE
Linear Regression Models Applied to Imperfect Information Spacecraft Pursuit-Evasion Differential Games

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Linville, Dax, A, Capt

5d. PROJECT NUMBER
20Y356G

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
Wright-Patterson AFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT-ENY-MS-20-269

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Labs, Space Vehicles Directorate
Dr. Alan Lovell, Dr. Andrew Sinclair
3550 Aberdeen Avenue SE
Kirtland AFB, NM 87117

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RV

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

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Statement A. Approved for Public Release; Distribution is Unlimited

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

14. ABSTRACT
Within satellite rendezvous and proximity operations lies pursuit-evasion differential games between two spacecraft. The extent of possible outcomes can be mathematically bounded by differential games where each player employs optimal strategies. A linear regression model is developed from a large data set of optimal control solutions. The model is shown to map pursuer relative starting positions to final capture positions and estimate capture time. The model is 3.8 times faster than the indirect heuristic method for arbitrary pursuer starting positions on an initial relative orbit about the evader. The linear regression model is shown to be well suited for on-board implementation for autonomous mission planning.

15. SUBJECT TERMS
relative satellite motion, differential games, optimal control, linear regression, clohessy-wiltshire, rendezvous and proximity operations

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Maj Joshuah Hess, AFIT/ENY
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	116	(937) 255-3636 x4713