7-2017

Optimal Control Methods for Missile Evasion

Ryan W. Carr

Optimal Control Methods for Missile Evasion

DISSERTATION

Ryan W. Carr, Major, USAF

AFIT-ENY-DS-17-S-055

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this dissertation 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-DS-17-S-055

OPTIMAL CONTROL METHODS FOR MISSILE EVASION

DISSERTATION

Presented to the Faculty
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy

Ryan W. Carr, BS, MS
Major, USAF

July 2017

AFIT-ENY-DS-17-S-055

OPTIMAL CONTROL METHODS

FOR MISSILE EVASION

DISSERTATION

Ryan W. Carr, BS, MS
Major, USAF

Committee Membership:

Dr. Richard G. Cobb
Chairman

Dr. Meir N. Pachter
Member

Major Scott J. Pierce, PhD
Member

Adedeji B. Badiru, PhD

Dean, Graduate School of Engineering Management

AFIT-ENY-DS-17-S-055

Abstract

Optimal control theory is applied to the study of missile evasion, particularly in the case of a single pursuing missile versus a single evading aircraft. It is proposed to divide the evasion problem into two phases, where the primary considerations are energy and maneuverability, respectively. Traditional evasion tactics are well documented for use in the maneuverability phase. To represent the first phase dominated by energy management, the optimal control problem may be posed in two ways, as a fixed final time problem with the objective of maximizing the final distance between the evader and pursuer, and as a free final time problem with the objective of maximizing the final time when the missile reaches some capture distance away from the evader.

These two optimal control problems are studied under several different scenarios regarding assumptions about the pursuer. First, a suboptimal control strategy, proportional navigation, is used for the pursuer. Second, it is assumed that the pursuer acts optimally, requiring the solution of a two-sided optimal control problem, otherwise known as a differential game. The resulting trajectory is known as a minimax, and it can be shown that it accounts for uncertainty in the pursuer's control strategy. Finally, a pursuer whose motion and state are uncertain is studied in the context of Receding Horizon Control and Real Time Optimal Control. The results highlight how updating the optimal control trajectory reduces the uncertainty in the resulting miss distance.

If a man will begin with certainties, he shall end in doubts;

but if he will be content to begin with doubts, he shall end in certainties.

-Francis Bacon, The Advancement of Learning

Acknowledgements

Dr. Richard Cobb, my advisor, deserves to be recognized for the large amount of work required to guide me through the process of obtaining a PhD. I counted 49 different slide presentations which he sat through patiently listening to my ideas. This doesn't include the many other formal and informal counseling sessions. I thank him for taking on this enormous commitment and seeing it through to the end.

My wife and children were often deprived of my full attention over the last few years, while I worked early or late to meet a deadline. I'm grateful they were willing to occasionally, temporarily, prioritize family time for work. I hope to repay them that time somehow.

Ryan W. Carr

Table of Contents

Page

Page

List of Figures

Figure Page

Figure Page

List of Tables

List of Abbreviations

Abbreviation Page

OPTIMAL CONTROL METHODS

FOR MISSILE EVASION

I. Introduction

When discussing aircraft maneuvers for missile evasion one should keep in mind several factors. First, events will transpire very quickly, and the aircraft operator will not be given the luxury of time to discover the best strategy available. Second, the operator may already be task-saturated, leaving a reduced mental bandwidth to deal with an additional, albeit important, priority. Finally, there will not be sufficient information available about the incoming threat to make the perfect decision, assuming it is even detected. The combination of these factors diminishes the pilot's ability to make use of all the traditional countermeasures to defeat threats. For this reason it may in some instances be beneficial to automate these processes. For some aircraft, the automation of existing countermeasures is already an operational reality. However, with the increased computational capabilities of newer aircraft, it is additionally possible to calculate and execute evasive maneuvers automatically. In the extreme case of automation, an Unmanned Aircraft System (UAS) would be responsible for all of these functions, and it then becomes the UAS designer's task to implement in software the aircraft's control response to a missile threat. However, even in the case of full automation, the same factors apply to the decision making process, meaning that the system must rapidly prioritize and act based on limited information.

1.1 Motivation

The Air Force Research Laboratory (AFRL) Strategic Vision for 2020 is to have "Intelligent machines seamlessly integrated with humans - maximizing mission performance in complex and contested environments." The Autonomy Science and Technology Strategy from Dec 2013 [1] highlights four strategic goals to guide AFRL to achieve its vision:

1. Deliver flexible autonomy systems with highly effective human-machine teaming
2. Create actively coordinated teams of multiple machines to achieve mission goals
3. Ensure operations in complex, contested environments
4. Ensure safe and effective systems in unanticipated dynamic environments.

Several keywords from these goal statements illustrate current technological challenges. The words *flexible*, *effective*, *safe*, and *coordinated* are used to describe these envisioned autonomous systems. These words indicate that a high degree of specialization is required for all subsystems, including the guidance, navigation, and control. Optimal control theory seeks a trajectory to maximize (or minimize) an objective subject to constraints. In the missile evasion scenario, the solution to an optimal control problem consists of the temporal inputs to the aircraft controls which will maximize the possibility of surviving a particular missile encounter, keeping in mind that an aircraft must obey the laws of motion and additionally must not perform any maneuvers which will endanger the pilot or structurally compromise the aircraft, for example by pulling very high g's. The careful construction of the optimal control problem is perhaps just as important as the solution. By crafting an appropriate objective and suitable constraints, the trajectory designer can create a guidance system that is flex-

ible, effective, and safe. For this reason, optimal control theory is an excellent tool to deal with missile evasion.

However, the solution will only be useful if the problem is posed correctly. Two questions then arise: what should be the objective, and what are the constraints? Many versions of pursuit-evasion problems have been solved in the literature, and these will be reviewed in detail in Chapter II. The outcome which is perhaps most closely linked with aircraft survivability is the miss distance, which is the range between the aircraft and missile at closest approach. For certain initial conditions, this moment occurs when the closing velocity passes from a positive value, through zero, to a large negative value. Because of this, an optimal evasion problem with final miss distance as the objective can be somewhat difficult to solve quickly and reliably because, at the instant of closest approach, the gradient of closing velocity is very high. It is therefore useful to pose the problem with a different objective, with either a fixed or free final time. For the fixed final time, one useful objective is to maximize the final range between the aircraft and the missile. For the free final time, another, related objective is to maximize the final time at which the missile intercepts the aircraft.

Regarding constraints, a majority of studies have focused on two-dimensional pursuit-evasion scenarios, primarily because closed-form solutions can more easily be obtained. Unfortunately, this leaves important features out of the analysis, namely the exchange of kinetic and potential energy that occurs as an aircraft ascends or descends. Another important but sometimes overlooked aspect is the dissipation and generation of energy by drag and thrust. The terminal phase of a missile evasion scenario requires the aircraft to perform highly dynamic maneuvers, which in turn generate a great deal of drag, depleting the energy state. Therefore the equations of motion, which serve as dynamic constraints in the optimal control problem, must

accurately model the generation and dissipation of energy via thrust and drag, and also the exchange between kinetic and potential energy through gravity.

There exists a relationship between the energy of a vehicle and its ability to maneuver, the subject of the aptly named Energy-Maneuverability theory [2]. Often a missile evasion scenario can be broken into at least two phases. In the first phase the exchange of energy is important, and the evader should seek to improve its energy state in relation to the missile's. During the second, or terminal phase of a missile evasion scenario, the aircraft will quickly exchange energy for maneuverability in an attempt to overwhelm the missile's ability to turn to achieve intercept. Therefore for the evader it is important that the aircraft begin the terminal phase with sufficient energy, and conversely, that the missile begin with as little as possible. This philosophy will motivate the form of the optimal control problems posed and solved within this work. Of course, the luxury to pilot the aircraft toward an ideal energy state prior to beginning evasive maneuvers may only apply to medium or long range encounters.

Referring again to the AFRL strategic goals, in describing the operational environment, the words *complex*, *contested*, *unanticipated*, and *dynamic* are applied. These hint at the uncertain nature of a battlefield. Despite modern Intelligence, Surveillance, and Reconnaissance (ISR) technology, lack of timely and coherent information remains a significant impediment to implementing missile countermeasures. Any proposed missile evasion strategy must acknowledge and quantify the information limitations, and inform the operator how to respond when new information is presented.

One specific method of dealing with limited information is to assume that unknown quantities must remain within specified bounds or possess a certain structure which may be informed by intelligence gathered on the missile or by expert opinion. Such a

strategy may be termed "Structured Uncertainty" and can be applied to the evasion problem by assuming that the missile may actively exploit this uncertainty to achieve its own objective. The resulting evasion trajectory will then be a guarantee on worst-case performance. Thus by admitting doubts about its adversary from the beginning, the evader may ensure its own survival. This is essentially the approach used in classic Pursuit-Evasion games, a subset of Differential Game Theory, wherein it is assumed that both the evader and pursuer may choose their controls, within some bounds, to optimize a performance objective. This mutual best response is known as an equilibrium solution. For a zero-sum game, where the two players have a single opposing objective, such as minimizing or maximizing the time to interception, the resulting solution is called a minimax or saddle-point.

To formalize the preceding discussion on the issues surrounding missile evasion, a hypothesis for this work is now proposed, along with corresponding research questions which will be answered in the body of this document.

1.2 Research Questions, Tasks, and Scope

Throughout this work on aircraft-missile pursuit-evasion, the aircraft will be known as the evader, while the missile will be called the pursuer.

1.2.1 Research Questions.

Hypothesis: A minimax aircraft-missile pursuit-evasion problem can be posed and solved to provide an open-loop control trajectory with a guaranteed cost despite uncertainty in the missile parameters, launch conditions, or guidance algorithm.

Research questions relating to this hypothesis are:

1. How should a medium or long range pursuit-evasion optimal control problem be posed and solved to increase the effectiveness of terminal maneuvers as defined in the previous section?

2. Can minimax solutions be found for medium or long range pursuit-evasion scenarios with realistic physical constraints?

3. How can optimal control solutions be found for medium or long range pursuit-evasion scenarios when the state and model parameters of the pursuer or evader are uncertain?

1.2.2 Research Scope.

While there are many methods available for solving optimal control problems (several of which will be briefly reviewed in Chapter II), the Pseudospectral Method for optimal control is particularly effective at solving problems with many states and controls. Because of the three dimensional nature of aircraft-missile evasion, it is necessary to represent each vehicle with at least six state variables (i.e. longitude, latitude, altitude, velocity, flight path angle, and heading) and three control variables (i.e. angle of attack, bank angle, and throttle). The large number of free variables makes analytical techniques and some computational techniques such as Dynamic Programming very difficult. Therefore, Direct Orthogonal Collocation, which will be fully described in Chapter II, is the tool primarily used in this research.

Two methods for calculating minimax trajectories will be highlighted here, the semi-Direct Collocation Nonlinear Programming (semi-DCNLP) and the Decomposition methods. These will be described in detail in Chapter II.

The content of the work is based on numerical simulation. Implementation of algorithms on real-time systems is a complicated endeavor that, while instructive in system integration and hardware application, would not greatly improve the theoret-

ical aspects of this work. For this reason the algorithms and techniques developed herein are not streamlined to function in real-time, although discussion of results may use computation time as a metric of suitability for a given method.

1.2.3 Research Tasks.

In order to address the above research questions, a number of tasks are accomplished.

1. Pose and solve a medium range aircraft missile evasion optimal control problem with final miss distance as the objective.

2. Pose and solve several alternative fixed and free final time medium range aircraft missile evasion optimal control problems and compare their performance versus the final miss distance objective in the following scenarios:

   (a) the adversary follows a prescribed guidance law.

   (b) the adversary implements an optimal strategy with uncertain parameters and initial states.

   (c) the uncertain state of the adversary is updated with periodic measurements.

1.3 Assumptions and Limitations

As has already been discussed, this work is entirely simulation based. It is assumed that with a significant amount of effort any of the methods presented could be implemented on a specific hardware configuration. Because that work would distract from the primary task of posing and solving unique optimal control problems, no attempt at hardware integration is made.

While many guidance systems exist for anti-aircraft missiles, such as remote control and beam riders, only homing guidance is considered here.

Although optimal control is often applied to discover inner-loop stability controllers, for this work it is assumed that sufficient controllers are already implemented, and the vehicles may be modeled as a point mass moving in the velocity and geographic coordinate frames. This eliminates the need to resolve a vehicle's orientation in the body axes, significantly reducing the computation required in the simulation by removing Euler angles, body axes rotation rates, body axes velocities, and control surface position and dynamics from the state vector. It is also assumed that the vehicles are able to coordinate their turns, removing the need to represent sideslip and side forces in the calculation of aerodynamic forces and the equations of motion. Finally, because most problems are of short duration, the effect of mass loss due to propellant burn will be assumed negligible for the aircraft (although not for the missile). The complete state vector and equations of motion of the aircraft and missile are fully described during the setup of each scenario.

1.4 Research Methodology

The primary tool used in this research is the Pseudospectral Method, a numerical technique for solving optimal control problems. This method is very general and can be applied to solve optimal control problems using a variety of constraints. As mentioned, the research tasks involve posing and solving a variety of optimal control problems related to aircraft-missile pursuit-evasion.

Specifically seven types of optimal control problems are posed and solved, although some variations of each are also presented in the document.

1. Problem 1: Closest Point of Approach (CPA) problem. The pursuer uses a suboptimal proportional navigation guidance scheme. The evader is given full

state information about the pursuer, and maximizes the distance between the pursuer and evader at the point of closest approach, i.e., when the closing velocity reaches zero.

2. Problem 2: Fixed final time, free final state (FX) problem. The pursuer uses the proportional navigation guidance scheme. The evader is given full state information about the pursuer, and maximizes the distance between the pursuer and evader at some fixed time prior to the terminal maneuver phase.

3. Problem 3: Free final time, fixed final state (FR). The pursuer uses proportional navigation guidance. The evader is given full state information about the pursuer, and maximizes the time when the range between the pursuer and evader reaches zero (or some capture radius).

4. Problem 4: Fixed final time, free final state, minimax (FXM). The pursuer and evader both have complete state information. The pursuer minimizes while the evader maximizes the distance between each other at a fixed final time prior to the terminal maneuver phase.

5. Problem 5: Free final time, fixed final state, minimax (FRM). The pursuer and evader both have complete state information. The pursuer minimizes while the evader maximizes the final time when the separation distance reaches zero (or some capture radius).

6. Problem 6: Receding Horizon Control (RHC). The pursuer uses proportional navigation. The evader has imperfect information about the pursuer's state, but receives updates and thus recalculates the FX trajectory at periodic intervals.

7. Problem 7: Real Time Optimal Control (RTOC). The pursuer uses proportional navigation. The evader has imperfect information about the pursuer's state, but receives updates and thus recalculates the FR trajectory at periodic intervals.

For reference throughout this document, Table 1 summarizes these scenarios. Each will be described in detail in the text.

Table 1. Optimal Control Scenarios

Problem	Acronym	Objective	Constraint
Closest Point of Approach	CPA	$\max_{u_E} r(t_f)$	$V_C(t_f) = 0$
Fixed Final Time	FX	$\max_{u_E} r(t_f)$	$t_f = Const$
Free Final Time	FR	$\max_{u_E} t_f$	$r(t_f) = Const$
Fixed Final Time Minimax	FXM	$\max_{u_E} r(t_f)$	$t_f = Const$
Free Final Time Minimax	FRM	$\max_{u_E} t_f$	$r(t_f) = Const$
Receding Horizon Control	RHC	$\max_{u_E} r(t_f)$	$t_f = Const$
Real Time Optimal Control	RTOC	$\max_{u_E} t_f$	$r(t_f) = Const$

1.5 Contributions

Specific novel contributions to the field of missile evasion and optimal control are documented in the body of this work. As a summary, they are:

1. Demonstrated that the fixed final time problem becomes the free final time problem as the fixed final time approaches the capture time (Chapter IV, Section 4.3).

2. Outlined a procedure to obtain an initial guess of the costates to use in the semi-DCNLP method (Chapter V, Section 5.2.2).

3. Proposed a penalty function for semi-DCNLP problems with a pure state constraint (Chapter V, Section 5.2.4).

4. Described an issue with using semi-DCNLP on certain free final time problems, and demonstrated the Decomposition method as an alternative solution technique (Chapter V, Section 5.3).

5. Demonstrated how the minimax solution represents a guarantee on the evader's performance despite uncertainty in the pursuer's model, guidance law, or initial state (Chapter VI, Section 6.1).

6. Developed an algorithm to improve the computational speed of complex RHC and RTOC problems by adjusting the mesh between each iterated solution (Chapter VI, Section 6.2.1.2).

1.6 Document Outline

A research hypothesis and related questions have been posed in this chapter, along with a list of tasks to be accomplished. Chapter II of this document surveys the existing open literature for methods which have already been investigated for optimal control and missile evasion. In Chapter III the vehicle models used in this research are described in detail. Chapter IV explores several one-sided optimal control solutions using both the fixed and free final time formulations. In Chapter V the two-sided, or minimax versions of the the fixed and final time problems are defined and solved. Chapter VI demonstrates results of using the fixed and free final time formulations in the presence of uncertainty.

Several appendices have also been included with this document. The first two detail adaptive meshing algorithms which, although not used directly in this work, were generated as byproducts while researching similar problems. The third appendix

provides instructions on setting up optimal control software on two microcomputers for real-time implementation.

II. Literature Review

2.1 Introduction

To organize missile evasion strategies, it is helpful to classify the problem by the assumptions made about the missile. Three types of pursuers will be the subject of study in this work. First, it will be assumed that the pursuer behaves according to a known control law such as proportional navigation, and that the state of the pursuer is known at all times. This will be known as the suboptimal pursuer. Second, it will be assumed that the pursuer's navigation law is unknown, although within specified boundaries, and that it behaves optimally. This will be called the optimal pursuer. Finally, it will be assumed that the initial state and vehicle parameters of the pursuer are subject to uncertainty. This will be the uncertain pursuer. While these three assumptions apply to the pursuer, it is easy to apply these assumptions to the evader as well. The goal of the current chapter is to describe methods which have previously been applied by other authors to model and solve problems for the three situations described above. To begin, a general overview of optimal control problems will be provided. This will be followed by a description of methods for setting up and solving optimal control pursuit-evasion problems with suboptimal, optimal, and uncertain adversaries. Throughout this document, the subscripts E and P will refer to the evader and pursuer, respectively.

2.2 Optimal Control Methods

Before discussing methods for calculating optimal control solutions, it is necessary to define the general type of optimal control problem to be solved. First it is assumed there exists a set of differential equations which model the motion of the evader

and pursuer. For the moment it will be assumed that the pursuer follows a known guidance law, making the problem one-sided. The dynamics may be written as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), \tag{2.1}$$

where $\mathbf{x}_E$ and $\mathbf{x}_P$ are the vector of state variables for the evader and pursuer, while $\mathbf{u}_E$ and $\mathbf{u}_P$ are controls for the evader and pursuer. For a one-sided problem, the pursuer's control follows an assumed guidance law, which usually depends only on the current state of both players. This is referred to as a perfect state feedback information pattern in the literature [3]. The guidance law serves to link the dynamics of the two together, forming a system of differential equations.

In the Bolza form of the optimal control problem, the objective or cost functional is defined with a Mayer part, ϕ, and a Lagrangian part, L, in continuous time as

$$J = \phi(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)dt. \tag{2.2}$$

The problem may also be subject to the boundary conditions

$$\boldsymbol{\Psi}(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f) = \mathbf{0}, \tag{2.3}$$

and the path inequality constraints

$$\mathbf{C}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}. \tag{2.4}$$

The goal is to find the control $\mathbf{u}(t)$, and in the process the state trajectory $\mathbf{x}(t)$, which will minimize the objective J. Many methods have been developed to solve this type of problem.

2.2.1 Indirect Method of Optimal Control.

Indirect methods utilize the calculus of variations to develop necessary optimality conditions which relate the optimal states and controls to optimal costates, $\boldsymbol{\lambda}(t)$. These conditions are best expressed by defining the Hamiltonian as

$$H(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\lambda}(t), t) = L(\mathbf{x}(t), \mathbf{u}(t), t) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t). \tag{2.5}$$

By appending the constraints to the cost function to form an augmented objective and then applying the calculus of variations, the necessary optimality conditions can be written as [4]

$$\begin{aligned} \dot{\mathbf{x}}^*(t) &= \frac{\partial H}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \boldsymbol{\lambda}^*(t), t) \\ \dot{\boldsymbol{\lambda}}^*(t) &= -\frac{\partial H}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \boldsymbol{\lambda}^*(t), t) \\ H(\mathbf{x}^*(t), \mathbf{u}^*(t), \boldsymbol{\lambda}^*(t), t) &\leq H(\mathbf{x}^*(t), \mathbf{u}(t), \boldsymbol{\lambda}^*(t), t), \end{aligned} \tag{2.6}$$

where the superscript * signifies the optimal state, control, or costate. The first two equations are called the state and costate equations. The third equation, known as Pontryagin's Minimum Principle, indicates that an optimal control must minimize the Hamiltonian. Boundary conditions are also developed depending on whether the problem is free or fixed final time and state. The system, a Two-Point Boundary Value Problem (TPBVP) with mixed boundary conditions, is typically difficult to solve for most problems due to the the necessity of calculating the derivatives of the Hamiltonian and the sensitivity and non-intuitive nature of the costates [5].

2.2.2 The Pseudospectral Method.

The difficulty of solving the TPBVP associated with the necessary conditions has led to a number of direct methods, where the continuous problem of Equations

(2.1)-(2.4) is transcribed to a discrete set of points. The dynamics are represented as a set of equality constraints enforced at the points, while the objective is calculated by some means of quadrature. The problem is then expressed as a Nonlinear Programming (NLP) problem and solved using one of a variety of well known NLP solvers [5–8]. While there are many ways to discretize the problem and form the NLP, recent application of Pseudospectral (PS) methods have proven fast and effective [9–11], and will therefore be highlighted.

In the PS Method, the optimal control problem is transcribed from a continuous problem to one satisfied at specific points. One convenient way to do this is by Lagrange interpolation. The state x may be approximated using Lagrange polynomial interpolation via the relation

$$\hat{x}(t) \approx \sum_{i=1}^{n+1} x_i L_i(t), \tag{2.7}$$

where the Lagrange polynomial basis is

$$L_i(t) = \prod_{\substack{j=1 \\ j \neq i}}^{n+1} \frac{t - t_j}{t_i - t_j}, \quad i = 1, ..., n+1. \tag{2.8}$$

The interpolation yields a polynomial approximation of the function which is exact at the points x_i. In between these points, the error is defined as [12]

$$x(t) - \hat{x}(t) = \frac{x^{n+1}(\xi(t))}{(n+1)!} \prod_{i=0}^{n} (t - t_i), \tag{2.9}$$

where the point ξ is the value of t where the $(n+1)^{\text{th}}$ derivative of the state x is equal to zero, $f^{n+1}(\xi) = 0$. By Equation [2.9], the error in a solution can be reduced by increasing the number of points. However, for many functions and discretizations this results in an unfavorable oscillation known as the Runge phenomenon, where

higher order derivatives cause the error to increase dramatically near the endpoints. One way to overcome this problem is to space the nodes closer to the endpoints than in the middle, such as by setting the nodes to be the roots of specific orthogonal polynomials, like the Legendre or Chebyshev.

Functions discretized via the roots of these polynomials also happen to have excellent error convergence properties in quadrature [13]. For the Legendre polynomials, Gaussian quadrature is exact for a polynomial of degree less than or equal to $(2N+1)$ points. While this result only applies to approximating the integration of polynomials, it gives confidence that choosing a Legendre polynomial basis for the point spacing will reduce error in general.

Collocating the optimal control problem via Lagrange polynomials onto a Legendre basis has only recently gained popularity in the field of optimal control [14]. Several varieties of the collocation are in use, which differ only by the exact version of the Legendre polynomial used. The points, τ_k, must be defined on the interval $-1 \leq \tau \leq 1$. The Legendre-Gauss (LG) points are the roots of $P_n(\tau)$, or the nth Legendre polynomial, which does not have points at -1 or 1. The Legendre-Gauss-Radau (LGR) points are the roots of $P_n(\tau) + P_{n-1}(\tau)$, which includes the point -1. The Legendre-Gauss-Lobatto (LGL) points use the roots of $\dot{P}_{n-1}(\tau)$, which include -1 and 1 [15].

An affine transformation shifts the problem from time, t, to τ, by

$$\tau = \frac{2t - (t_f + t_0)}{t_f - t_0}, \tag{2.10}$$

and is bounded by $-1 \leq \tau \leq 1$ as required. Because the LG and LGR points are missing endpoints, an additional, "non-collocated" $\tau_{n+1} = 1$ point is typically included.

Once the optimal control problem has been discretized using one of the above methods, it is possible to calculate an accurate approximation of the integration and derivative operators. The derivative of the state at each discretized τ_i point is found using the differentiation matrix, $\mathbf{D}$, which is equivalent to the rate of change of the Lagrange polynomial,

$$\dot{x}(\tau_i) \approx \sum_{k=1}^{n+1} x_k \dot{L}_k(\tau_i) = \sum_{k=1}^{n+1} \mathbf{D}_{ki} x_k. \tag{2.11}$$

The differentiation matrix is distinct for each collocation scheme being used. For the LG method it is an (n x n) matrix, while for the LGR the matrix is (n x $n+1$). Using the LGR differentiation matrix to calculate the derivative or integral of a state or control is considerably more accurate than using fixed interval methods (such as the 3 or 5 point formulas) because it uses information from every discrete point, i.e., it is a global method [16]. The differentiation matrix is a linear operator, and the discretized state derivative may be expressed in matrix multiplication form as

$$\dot{\mathbf{x}} \approx \mathbf{D}\mathbf{x}. \tag{2.12}$$

Referring to Equation (2.1), the dynamics of the problem represent an equality constraint on the optimal control problem. Using the differentiation matrix, the relation becomes

$$\mathbf{D}\mathbf{x} - \mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{0}, \tag{2.13}$$

where $\mathbf{x}$ and $\mathbf{u}$ represent the discretized state and control vectors at every collocation point. The constraint is enforced at each collocation point, creating n equality constraints associated with the dynamics for each state.

The quadrature weights, w_k, are associated with the collocation scheme, and approximate the integration of the running cost, L, as [17]

$$\int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)dt \approx \frac{t_f - t_0}{2} \sum_{k=1}^{N} w_k L(\mathbf{x}(\tau_k), \mathbf{u}(\tau_k), \tau_k). \tag{2.14}$$

Thus the cost function is also collocated, and together with the dynamic equality constraints form an NLP problem. Additional constraints are imposed for state and control limits, initial or final conditions, time constraints, and path constraints. This is done by setting an equality or inequality constraint at the relevant collocation point. PS schemes have been implemented into software by several groups [18–20]. Due to the pseudospectral convergence achieved by these solvers, they deliver highly accurate solutions, and generally require relatively low computation time. Depending on the NLP solver used they may benefit from sparse matrix operations, further increasing computation speed.

Unfortunately there are a few limitations associated with the most common NLP solvers. SNOPT [6] is a Sequential Quadratic Programming (SQP) algorithm wherein the NLP is solved using a sequence of quadratic programming subproblems. The algorithm uses first derivative information stored in the Jacobian matrix with a quasi-Newton method known as BFGS to recursively approximate second derivative information stored in the Hessian matrix [21]. The quality of a given solution depends on several user defined parameters, but most notably the NLP error tolerance. Because this method depends on these local gradients, it is sensitive to the initial guess, which sometimes results in the solver not finding a solution, or requiring a large number of iterations to discover one. Furthermore, when a solution is achieved it should be not be considered a global optimum. In fact, because the direct collocation method only achieves the first-order necessary conditions, solutions should only be considered as candidate extrema until sufficiency is demonstrated.

While many versions of PS optimal control software exist, the tool used in this work is the General Purpose Pseudospectral Optimal Control Software-II (GPOPS-II)

[18]. It is based in MATLAB, uses the LGR collocation scheme employing differential and integral forms [15, 17], and offers a variety of user options for different *hp* meshing methods [22–24] in order to reduce the discretization error. The user is required to define a tolerance for this error, which differs by the method being used. The derivative information passed to the NLP solver is approximated using a sparse forward, central, or backward finite-differencing perturbation method [25]. The software exploits sparsity by detecting dependencies in the constraint Jacobian passed to the NLP software using MATLAB's "not-a-number" (NaN) representation. This is accomplished by passing a NaN into the constraint function for a specific test variable, and allowing it to propagate into the output. Any NaN detected signifies that the output is dependent on the tested input, whereas any output with a real value is not dependent. A GPOPS-II problem can be broken into phases, where subproblems can be implemented and linked via additional constraints. This allows flexibility in dealing with diverse dynamics within a problem unified by a single objective function.

2.2.3 Costate Estimation.

Early researchers in PS methods were interested in estimating the costates of the optimal control problem. Although the costates are not required in order to obtain solutions, it was felt that by showing that the necessary optimality conditions were satisfied, the solution obtained using the direct method was valid. Gong et al. [26] showed that, given the correct mapping between the discretized costates and the Karush-Kuhn-Tucker (KKT) multipliers resulting from the solution of the NLP, the costates from the direct method converged to the costates from the indirect method. This mapping enables estimation of the continuous costates from the solution of a PS

optimal control problem. For the LGR collocation scheme, the mapping is [17]

$$\begin{aligned} \boldsymbol{\lambda} &= \mathbf{W}^{-1}\boldsymbol{\Lambda}^{LGR} \\ \boldsymbol{\lambda}_{N+1} &= \mathbf{D}_{N+1}^{T}\boldsymbol{\Lambda}^{LGR}, \end{aligned} \tag{2.15}$$

where $\boldsymbol{\lambda}$ is a vector of the continuous costates, $\mathbf{W}$ is a diagonal matrix of quadrature weights, $\boldsymbol{\Lambda}^{LGR}$ is the vector of KKT multipliers, and $\mathbf{D}_{N+1}$ is the $N+1$ column of the LGR differentiation matrix. Fahroo and Ross note that, at least for their discretization scheme (LGL), the costate estimates have the same order of accuracy as the states [27].

2.2.4 Indirect Transcription.

The PS method is usually thought of as a direct method. However, it is possible to use collocation methods to solve problems indirectly. Betts describes the Indirect Transcription Method [5] where the necessary conditions for optimality are expressed as a TPBVP which is collocated and transcribed to a NLP problem. To use this method using a PS solver like GPOPS-II, it is necessary to include the costate equations as additional states, replace the controls with the stationarity conditions, apply terminal costate conditions as endpoint constraints, and set the objective function to zero. Betts describes the direct transcription method as "discretize then optimize", and the indirect transcription method as "optimize then discretize."

He also discusses some of the difficulties associated with this method. First, when state constraints are present the problem must be broken into phases of constrained and unconstrained arcs, along with jump conditions on the costates. Prior to solving the problem it is unknown exactly how many subarcs are present, and thus the user must make a guess. A second complication is the nonintuitive nature of the costates, complicating the initial guess. The costate dynamics are often very sensitive to the

initial guess, meaning that even for a relatively good initial guess, the numerical solution may be ill-conditioned. Finally, this method requires an expert user, and complicated problems will likely be impossible to even formulate, let alone solve.

2.3 Suboptimal Strategies

Before studying optimal strategies for evasion, it is important to review likely sub-optimal guidance algorithms which may be applied by a pursuer. The most common and widely studied version of homing missile guidance is Proportional Navigation (PN).

2.3.1 Proportional Navigation.

The field of missile guidance has existed since World War II, with the advent of modern rocketry. Early German missiles were used against ground targets and relied on inertial guidance, resulting in somewhat poor accuracy by today's standards, but enough to target a large city. Inertial guidance is insufficient when dealing with a maneuvering target whose future coordinates are unknown. The U.S. Navy benefited from German research on PN, a guidance technique still used in homing missiles today [28]. They began to develop a homing missile system using continuous wave radar known as the "Lark", and after six years of development and testing achieved a successful intercept. Many technological advances were required to achieve this feat in radar filtering, electronics reliability, and hardware-in-the-loop simulation.

Throughout the 1950s the advances came more quickly. Some new missiles used semiactive radar, where the transmitter remained on the ground while the receiver traveled with the missile. The first infrared (IR) version of the Falcon missile went operational in 1956 [29]. Further advancements made in propulsion, controls, and radar improved missile expectations about performance and several systems were op-

erational by the end of the decade. Unfortunately combat experience in Vietnam showed that missiles did not perform as expected, due to assumptions about their operational use. Two reasons are cited by one author [30], the first being that while missiles were carried under aircraft almost daily, sometimes months went by before the missiles were actually used. This operational stress degraded the reliability of the missile so that often the system did not work. The second reason was that designers assumed that missile launches would occur against low maneuvering targets (bombers) beyond visual range. However, policy required that pilots visually identify targets, so that launch ranges were shorter than the missile was intended, resulting in large required acceleration commands. Unfortunately the maximum target acceleration for which the missiles were designed was only 3 g. Interestingly, the author mentions that one solution to this last problem was to change the value of a few resistors, increasing to 15 g the electronic limit artificially imposed upon the control system, enabling a higher missile maneuverability.

Modern missile technology continues to improve systems by widening the sensor's field of view, enhancing resistance to countermeasures, increasing tracking sensitivity, and boosting maneuverability [29, 30]. However, the principal guidance strategy is still most commonly based on PN [31].

After pure pursuit navigation, where the pursuer's heading is selected to be the current evader's position, PN is the most natural and intuitive pursuit strategy. Any sailor knows how to intercept another moving craft by maintaining a constant bearing. This strategy, known as constant bearing navigation, collision course navigation, or parallel navigation, has been used for centuries, and has even been demonstrated in animals such as dragonflies [32], bats [33], dogs [34], worms [35], and baseball players [36]. The PN guidance law seeks to attain this type of navigation by accelerating the

missile perpendicular to the line of sight, with magnitude proportional to the line of sight rate. For motion restricted to the horizontal plane, this can be written as

$$n_c = N\omega, \tag{2.16}$$

where n_c is the acceleration command (m/s^2), N is the navigation ratio (m/s), and ω is the time rate of change of the Line of Sight (LOS) angle from the pursuer to the evader (rad/s). The navigation ratio is sometimes expressed as the product of some velocity and an effective navigation ratio, N_P. The velocity used in this relation depends on the variety of PN being implemented. While the navigation ratio may vary with time, the effective navigation ratio typically takes on constant values between 3 and 5 in order to avoid large acceleration and control saturation as the pursuer nears the evader [30]. The PN guidance law can be proven as the optimal method to intercept with minimal control given the assumptions that [37]:

1. The kinematics are linear,
2. full state feedback is available,
3. the evader and pursuer speeds are constant,
4. the evader is non-maneuvering,
5. the missile responds instantaneously to acceleration commands (the missile time constant is zero).

While these assumptions may seem limiting, it should be noted that they describe restrictions on the optimality of PN, not its general effectiveness. Deviations from these assumptions while using PN may still result in an intercept, but it may require more commanded acceleration and time than in an idealized scenario.

In the literature there exist many distinct implementations of PN. A few notable versions are Pure Proportional Navigation [38, 39], True Proportional Navigation [40, 41], Generalized Proportional Navigation [39, 42], and Idealized Proportional Navigation [43]. Each of these differ by the exact way in which they implement Equation (2.16). In the current work, the PN version presented by [44] and [45] will be implemented. This requires calculation of the closing velocity, V_C, which is the negative rate of change of the distance between the pursuer and evader, given by

$$V_C = -\frac{\mathbf{r} \cdot \dot{\mathbf{r}}}{|\mathbf{r}|}. \tag{2.17}$$

Above, $\mathbf{r}$ is the relative inertial position vector from the pursuer to the evader and $\dot{\mathbf{r}}$ is the relative inertial velocity vector between the two. The rate of change of the LOS between the pursuer and evader is

$$\boldsymbol{\omega} = \frac{\mathbf{r} \times \dot{\mathbf{r}}}{\mathbf{r} \cdot \mathbf{r}}, \tag{2.18}$$

which is composed of three elements ω_x, ω_y, and ω_h in the inertial East-North-Up (ENU) reference frame. These components can be projected onto the missile's pitch and yaw axis by

$$\begin{aligned} \omega_\gamma &= -\omega_x \sin\chi_P + \omega_y \cos\chi_P \\ \omega_\chi &= \sin\gamma_P(\omega_x \cos\chi_P + \omega_y \sin\chi_P) + \omega_z \cos\gamma_P, \end{aligned} \tag{2.19}$$

where χ_P is the pursuer's heading and γ_P is the pursuer's flight path angle.

Finally, the longitudinal and lateral accelerations commanded to the missile are

$$\begin{aligned} a_{\gamma,C} &= N_P V_C \omega_\gamma + g\cos\gamma_P \\ a_{\chi,C} &= N_P V_C \omega_\chi. \end{aligned} \tag{2.20}$$

2.3.1.1 Time-To-Go and Zero Effort Miss.

Two additional concepts are useful to analyze pursuit-evasion engagements. The time-to-go (t_{go}) is the amount of time before the pursuer reaches the closest point of approach. This value is often used in guidance algorithms, and thus it is desirable to generate a reasonable estimate. The time-to-go is not known precisely, given that there is no guarantee that an interception will actually occur, and that the evader will likely maneuver to delay this event. However, estimates may be made based on assumptions of the pursuer's dynamics and guidance laws, and predictions about the evader's future behavior. The simplest estimate is

$$t_{go} = \frac{|\mathbf{r}|}{V_c}. \tag{2.21}$$

This estimate assumes that both evader and pursuer are traveling at constant velocity without maneuvering, and that an interception will occur. A slightly better estimate can be obtained by evaluating the derivative of the square of the linearly propagated range, setting to zero, and solving for time [46]. This approach gives

$$t_{go} = -\frac{\dot{\mathbf{r}} \cdot \mathbf{r}}{\dot{\mathbf{r}} \cdot \dot{\mathbf{r}}}, \tag{2.22}$$

This expression is slightly more accurate because it estimates the moment of closest approach, which works for a miss or an intercept.

More sophisticated estimates are abundant in the literature [46–48]. Some authors solve a separate optimal control problem, minimizing range or final time [49, 50]. One specific reason for estimating an accurate t_{go} is that Equation (2.16) can be reformulated as [28, 31]

$$n_c = N' \frac{ZEM}{t_{go}^2}, \tag{2.23}$$

with the Zero Effort Miss (ZEM) as the distance the pursuer would miss the evader if both continued along their present course without acceleration. In three dimensions it may be considered as the predicted interception coordinate toward which the pursuer should steer (hence the form of Equation (2.23)). In three dimensions, the components of the ZEM are calculated by

$$\mathbf{ZEM} = \mathbf{r} + \dot{\mathbf{r}} t_{go}. \tag{2.24}$$

In analytic studies of PN, there are several other parameters of interest. One of these is the capturability region, defined as the bounds of range and LOS angle for which if a missile is launched, an interception of a non-maneuvering target can be guaranteed. Additionally of interest in analytic studies are the interception angle θ_f and the interception time t_f.

2.3.2 Aircraft Survivability and the Miss Distance.

Aircraft survivability is a broad topic, but the essential can be expressed with a single number, the Probability of Kill, P_K. In an encounter with a single missile, this number represents the likelihood that the evader will be incapacitated or destroyed by the pursuer. It can be broken down into Susceptibility, P_H, which is the probability of being hit, and Vulnerability, $P_{K|H}$, the probability of being killed given a hit [51]. This relationship is given as

$$P_K = P_{K|H} P_H \tag{2.25}$$

Reducing the Vulnerability of the aircraft is primarily accomplished during the aircraft design stage, and therefore not directly useful here. However, the Susceptibility of the aircraft can be reduced by adopting countermeasures. This research

will not address the many and useful countermeasures available for modern aircraft, but instead will focus on the one countermeasure available to all pilots: evasive maneuvering. The susceptibility of an aircraft to being hit is difficult to represent by a simple equation, since it depends on the guidance system of the missile, the fusing mechanism, the orientation of the aircraft and missile, the fragmentation pattern of the warhead, and of course the tactical maneuvering used by the pilot.

However, it is possible to perform a simple analysis by defining the miss distance plane of the aircraft by the vector connecting the closest point of approach to the centroid of the aircraft, and normal to the propagated fragments of the detonated munition. In this frame, the centroid of the aircraft is the origin, and points where propagated fragments pierce the aircraft are described by the Cartesian coordinates (ξ, ζ). If it is assumed that the ξ and ζ components are uncorrelated, and that their error is represented by a bivariate normal distribution, then the probability density function of a hit is given by

$$\eta(\xi,\zeta) = \frac{1}{2\pi\sigma_\xi\sigma_\zeta}\exp\left[-\frac{(\xi-\mu_\xi)^2}{2\sigma_\xi^2} - \frac{(\zeta-\mu_\zeta)^2}{2\sigma_\zeta^2}\right] \tag{2.26}$$

where μ_ξ, μ_ζ, σ_ξ, and σ_ζ are the mean and standard deviation of the propagated fragments. The magnitude of the miss distance is the magnitude of the vector (μ_ξ,μ_ζ). The probability of hit is then given by

$$P_H = \int_L\int \eta(\xi,\zeta)d\xi d\zeta, \tag{2.27}$$

where the integration is taken over the irregularly shaped surface of the aircraft. While this integral may be complicated to calculate, it is clear that for a large miss distance, $|(\mu_\xi,\mu_\zeta)|$, the integral becomes small, and thus the probability of hit is reduced. This discussion reveals that one certain way to decrease the P_H, and thus improve the

survivability of the aircraft, is to maximize the final miss distance. Calculation of actual values for P_H would require a specific aircraft geometry and missile warhead to be adopted, and thus will not be discussed in this work due to the sensitive nature of the numbers that would be generated.

2.3.3 Evasion Strategies.

When faced with a pursuer who adopts a suboptimal strategy such as PN, the evader can predict the behavior of the pursuer and exploit this knowledge to maximize the probability of evasion. This immediately suggests posing an optimal control problem with an objective and constraints, although what these should be is the subject of many studies. In order to make sense of the numerous works that have been published, it is useful to provide a taxonomy of optimal control problems used in pursuit-evasion publications. One convenient way to label optimal control problems is simply by whether they have a fixed or free final time. In a fixed final time problem (FX) the dynamics are propagated until some predetermined time value is reached. The final state may be constrained in some way or it may remain completely open. In a free final time problem (FR), there must be a constraint on the state or on some combination of states in order for the problem to be tractable.

Differential game theory encompasses optimal control theory. In a differential game multiple players are allowed to behave independently, whereas in an optimal control problem there is either only one player, or else the multiple players act collaboratively. A two-person non-cooperative pursuit-evasion differential game can be converted into an optimal control problem by assuming a suboptimal strategy for either the pursuer or evader, and revealing this information to their adversary. Two concepts can be borrowed from differential game theory to further aid in classifying the numerous combinations of objective and constraints. When a problem is posed

with only specific outcomes, such as determining whether or not the pursuer intercepted the evader, it is called a game of kind [52]. Alternatively, a problem may be posed where the solution space is continuous, such as maximizing the distance between the evader and pursuer at some instant in time. This is called a game of degree. Often, in games of pursuit, it is helpful to solve a game of degree within a game of kind, for example by first assuming that interception occurs then attempting to establish the time required to complete the game.

The concept of game of kind versus game of degree appears in optimal control problems as the end constraints versus the objective. The end constraints define the conditions for obtaining a solution, while the objective describes how well the solution meets the goal. For example, one might prefer to establish the result of the game of kind by dictating that the pursuer will be assumed to always intercept the evader (as an end constraint). Then, the game of degree may be to maximize the time required for intercept (as an objective). In constructing this problem it is assumed that although the simulated pursuer achieves intercept, the evader's chance of survival would be increased during an actual pursuit-evasion encounter due to the large amount of time achieved by maximizing the objective. Alternatively, it can be assumed that interception does not occur, by posing the problem with a fixed final time constraint during which the pursuer cannot possibly achieve interception (game of kind). Then the objective may be to maximize the final range between the pursuer and evader or to minimize the final closing velocity (game of degree). Both the constrained final time and constrained final state type of problems can be described by their problem of kind and degree. A number of problems have been proposed in the literature, and they will be classified here.

2.3.3.1 Constrained Final Time.

When the final time is constrained, it is usually assumed that the pursuer will not achieve intercept within the prescribed time limit, meaning that the problem of kind assumes no intercept. This does not mean that the pursuer would not have achieved intercept in the future, just that the problem is only defined up until a certain time. In this type of problem the evader may take action to enhance its survival by one of several methods. Some authors choose to maximize the range between pursuer and evader at a fixed final time [53–56]. The fixed final time may be chosen based on the time-to-go parameter calculated at the initial time, although this is only an approximation. The choice of trying to maximize range is logical given that the evader wishes to prolong interception; however, it leaves the problem incomplete in the sense that the missile is still in pursuit of the evader.

Another interesting option is to maximize the maximum line of sight rate attained before the final time [57]. The author argues that because the pursuer is typically faster and more nimble than the evader, and has higher load limits, it does not make sense to attempt to outrun or outmaneuver the pursuer. Instead, by attempting to maximize the line of sight rate, it may be possible to saturate the tracking sensor being used by the pursuer, thus causing the pursuer to lose lock. This tends to result in trajectories where the evader holds a steady course, possibly directly toward the missile, dodging at the precise moment in order to maximize the line of sight rate. In order to achieve such a dynamic escape, it must be assumed that the evader can accurately track the state of the pursuer. Otherwise it would be difficult to determine the exact moment when the evasive maneuver must be applied.

In a recent study, the author proposed to minimize the integral of the two norm of the control of the evader, i.e. the control energy, while constraining the miss distance to reach a minimum acceptable value at the final time [58]. The final time is set as

the interception time predicted by the time-to-go calculation from Equation (2.21). By assuming linear dynamics and complete state information, the authors derive a guidance law similar similar to PN. The benefit of minimizing control effort is to reduce speed losses due to induced drag, which may be useful in practical applications.

2.3.3.2 Constrained Final State.

When the final time is left free, it is necessary to impose final constraints on the state, or some combination of the states, in order to properly define an end condition for the optimal control problem. There are many interesting combinations of end constraints and objective functions. One particular study outlines and compares several optimal control problems for missile evasion, and they have been summarized here [44].

Table 2. Objectives and end constraints for free final time optimal evasion problems.

#	objective	end constraint	reference
1	maximize t_f	$r(t_f) = r_f$	[44]
2	minimize V_c	$r(t_f) = r_f$	[44, 59]
3	maximize $r(t_f)$	$V_c(t_f) = 0$	[44, 45, 60–62]
4	maximize $\int_{t_0}^{t_f} a(t)dt$	$r(t_f) = r_f$	[44]
5	maximize $\sigma(t_f) - \chi_p(t_f)$	$r(t_f) = r_f$	[44]
6	maximize $\omega(t_f)$	$r(t_f) = r_f$	[44]

In Table 2, t_f is the final time, r is the range between the pursuer and evader, r_f is a predefined capture radius, such as the blast radius of a missile, V_C is the closing velocity between the pursuer and evader, a is the total control effort of the pursuer (a positive number), σ is the LOS angle, ω is the angular tracking rate or rotation of the LOS vector, and χ_P is the heading of the pursuer.

One problem with this method is that it assumes that the end constraints will be met, imposing a problem of kind, which may sometimes cause one of the players to

act illogically in order to achieve the end constraints. For example, if the objective is to maximize time to capture, while the end constraint is set that the final range must be equal to zero, and the evader is faster than the pursuer, it is possible to determine initial conditions which would not end in a capture. Thus normally the problem has no solution for those initial conditions. However, in an attempt to find a feasible solution, the optimizer will send the evader back toward the pursuer at the cost of reduced capture time, a counterintuitive behavior. For this reason it is important to avoid conditions where the problem setup forces the evader to make an unrealistic decision in order to satisfy the end constraints.

Entry number three in the table is perhaps the most widely used in the literature. It assumes that the pursuer is initially faster than the evader. While the pursuer closes in on the evader the closing velocity, V_C, will be positive. If the pursuer intercepts the evader, $r(t) = 0$, then the closing velocity will be exactly zero. If the evader is able to dodge the pursuer, then the closing velocity will become negative. The closing velocity must pass through zero at this time, and the distance between the pursuer and evader is the terminal miss distance, $r(t_f)$. An evader desires to maximize this distance, while a pursuer wishes to minimize it. This combination of objective and constraint does not inappropriately impose a solution to the game of kind, since it admits both intercept and evasion. The terminal miss distance is also the dominant factor in aircraft survivability [63], thus it is a logical choice for the objective of a short range optimal control problem. This combination of objective and constraints, here called the CPA problem, will be studied in detail in Chapter IV.

2.3.4 Energy-Maneuverability Theory.

It is useful to regard an aircraft as a mechanism for the exchange of energy. According to the law of conservation of energy, an aircraft possesses a certain amount

of potential energy at altitude, and with the help of gravity this potential can be converted into kinetic energy by diving. Additionally, the system will gain energy from the engine thrust, and lose energy due to drag. Rutowski used this knowledge to calculate climbing trajectories which are optimal in terms of minimum time and minimum fuel expenditure in a classic paper [64]. In the paper, he formulates the current energy state of the aircraft as

$$E = Wh + \frac{WV^2}{2g}, \tag{2.28}$$

where W is the weight of the aircraft, h is the altitude, V is the velocity, and g is the acceleration due to gravity. This equation represents the exchange of potential and kinetic energy. As an aircraft climbs, it gains potential energy, but typically at the expense of kinetic energy. A more convenient parameter which does not change with fuel consumption is simply the total energy divided by the weight of the aircraft, or the specific energy,

$$E_S = h + \frac{V^2}{2g}. \tag{2.29}$$

Taking the derivative of E_S with respect to time one has

$$\frac{dE_S}{dt} = \frac{dh}{dt} + \frac{dV^2}{dt}\frac{1}{2g} = \frac{dh}{dt} + V\frac{dV}{dt}\frac{1}{g}. \tag{2.30}$$

Assuming the aircraft behaves like a point mass gives the time rate of change of velocity and altitude to be

$$\begin{aligned} m\frac{dV}{dt} &= T\cos\alpha - D - mg\sin\gamma \\ \frac{dh}{dt} &= V\sin\gamma, \end{aligned} \tag{2.31}$$

where m is the mass, T is the thrust force, D is the drag force, α is the angle of attack, and γ is the flight path angle. Substituting these into Equation (2.30) gives

$$\frac{dE_S}{dt} = V \sin\gamma + V(T\cos\alpha - D - mg\sin\gamma)\frac{1}{mg} \tag{2.32}$$

Applying the quasi-steady assumption [65] that the angle of attack is small leads to the equation for the time rate of change of the specific energy, which is equivalent to the specific excess power

$$P_S = \frac{dE_S}{dt} = V\frac{T-D}{W}. \tag{2.33}$$

This equation highlights that the thrust is the source of energy generation, while the drag is a dissipation term. Equations (2.29) and (2.33) have been used to develop optimal trajectories for rapidly ascending to a specific altitude and velocity, minimizing fuel required for a climb, maximizing range for a given throttle setting, and minimizing range in a glide [64, 65].

Rutowski explained that the minimum time to climb can be achieved by flying a trajectory such that the aircraft dives along constant lines of specific energy to increase the velocity, followed by a climb along the points of maximum excess power for a given specific energy. Graphically this is the set of points where the contour lines of excess power are tangent to the lines of specific energy. It is likely that during steep dive and climb portions the assumption of small angle of attack is violated, which means that the actual optimal path would stray from the line of constant specific energy. Also, the lines of constant P_S are calculated for 1 g, so that large aerodynamic loads in a dive or climb would not exactly fit the chart. However, the overall idea of using the known energy profile of the aircraft can serve as a guide to pilots in many situations.

This fact was noted by a fighter pilot named John Boyd in the mid 1960's. He indicated that the ability of an aircraft to maneuver is linked to both its current energy state, and also how well that energy state is managed for subsequent actions [2]. Boyd theorized that during an encounter with an adversary, a pilot should either be at a higher energy state than an opponent, or be able to attain energy more quickly. He used the concepts developed by Rutowski to calculate Mach-altitude diagrams for a variety of then current fighter aircraft. By studying these diagrams, it was possible to suggest best paths for gaining energy quickly during an encounter, thus putting a pilot at an advantage over an adversary. These H-M (altitude vs Mach) diagrams could also be used to compare the performance of two aircraft. By superimposing two H-M diagrams it was easy to identify when an aircraft had a P_S advantage, meaning that at a certain Mach number and altitude it would be able to gain energy more quickly, and in turn convert this energy into superior maneuverability.

Boyd discussed two other important parameters for maneuverability, the maximum turn rate and the load factor, n, which are commonly displayed in the "dog-house" plot and the V-n diagram, respectively. Here, the ability of an aircraft to turn, represented by either the maximum turning rate or the aerodynamic force relative to the weight of the airplane, or g, is plotted versus velocity for a series of altitudes. The number of g an aircraft is able to pull is limited at low velocities by the stall limit of the aircraft, and at high velocities by human and structural limitations. These diagrams contain an important point, known as the corner velocity, where the aircraft achieves its highest turn rate because at lower velocities the aircraft cannot risk pulling more g due to stall. The rate of change of heading is inversely proportional to the velocity, so that at higher velocities the maximum turn rate is decreased. Therefore it is important to understand where this maximum turn rate occurs, and how it plays into the the maneuverability of the aircraft. Boyd suggested that this diagram

represents the instantaneous maneuverability, while the H-M diagram represents the sustained maneuverability, or the ability to exchange energy for maneuverability.

Boyd also suggested that these diagrams be used to plan pre- and post-engagement maneuvers, to maximize the ability to quickly generate energy, and restore it after the action. He indicated that the theory does not necessarily change the dogfighting tactics already in use, but simply augments the planning of how and when to implement those tactics in order to achieve an advantage before and after the encounter. Although this work focused mostly on dogfighting, it may be interesting to regard missile evasion from the standpoint of Energy Maneuverability (EM) theory.

A medium-range pursuit-evasion scenario may have multiple phases, depending on the relative position, heading, and energy state of the pursuer and evader. In one study [66], three distinct regions were identified, differing by the strategy used by the evader. The first region is defined by a short duration engagement where the evader does not have sufficient time to fully implement any strategy. This is termed the Low Effectiveness Zone (LEZ). In the second region the evader just has time to apply the necessary turn to dodge the pursuer. In this region maneuverability is dominant. This is called the Low Sensitivity Zone (LSZ) because the evader does not have time to influence the final miss distance beyond the ability to turn rapidly. In the third region, the evader has time to set up for the final maneuver, and does this by attempting to reduce the closing velocity. This region is termed the High Sensitivity Zone (HSZ) and is dominated by aerodynamic drag. A visual depiction of these zones, adapted from reference [66], is shown in Figure 1.

These three zones fit well within the scope of analysis techniques described by EM theory. In the study, it states that in the HSZ, the evader should attempt to minimize the closing velocity. However, the study only considered two dimensions, meaning that the exchange between potential and kinetic energy could be an important factor

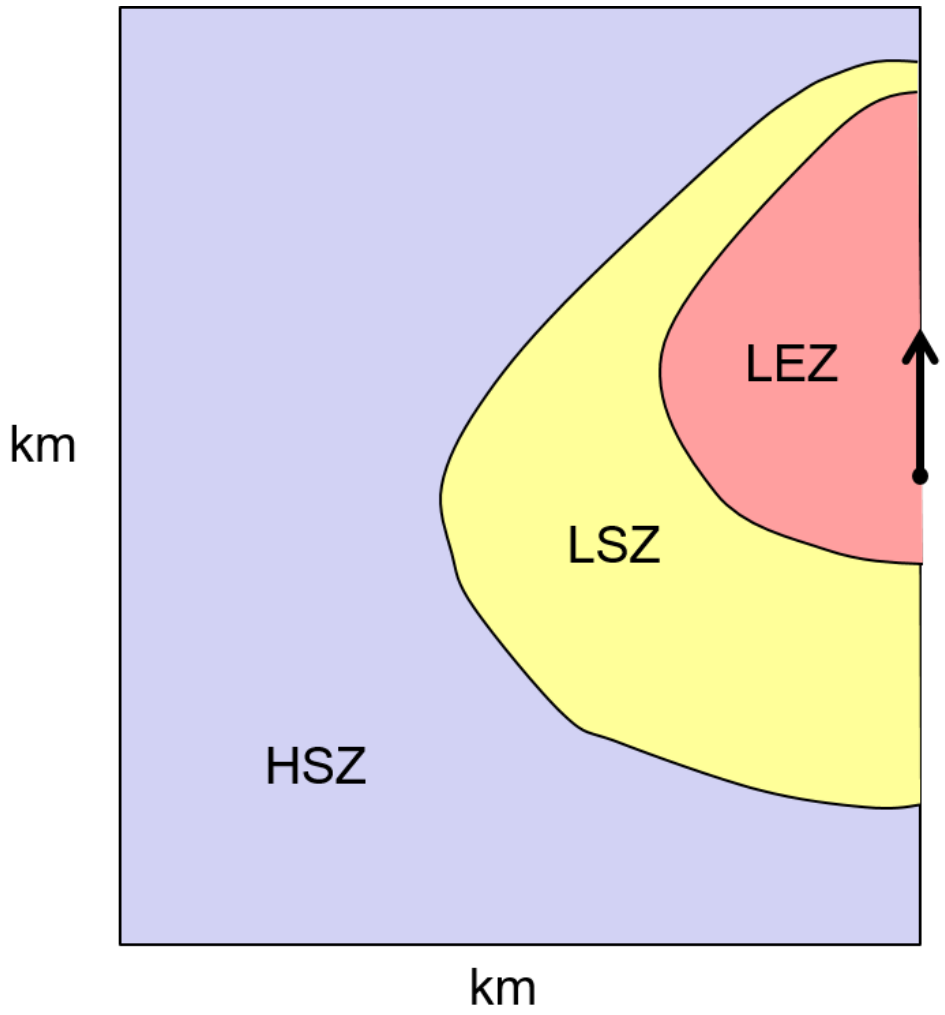

Figure 1. A visual depiction of the Low Effectivenss Zone (LEZ), the Low Sensitivity Zone (LSZ), and the High Sensitivity Zone (HSZ) from Shinar and Tabak.

in this region. Clearly, in the LEZ, it is desirable to maximize maneuverability, which is achievable by entering the zone with sufficient specific energy.

Energy has been used as a parameter in medium-range missile guidance research. One study substituted velocity with energy as a state in the equations of motion, and used a combination of the missile's total energy and terminal time as the objective against a stationary target in an optimal guidance problem [67]. Another study maximized the terminal velocity of the missile, which at a given altitude is equivalent to maximizing the specific energy [68]. The missile was guided to a fixed point in space

under the assumption that the ground support system could predict an appropriate interception. The authors showed that for longer ranges, using the velocity of the missile as an objective prior to the homing phase is favorable, while for short ranges, interception time seemed to be a better objective.

2.4 Optimal Strategies

As has been discussed, PN is the optimal guidance for a pursuer to achieve intercept with minimum control given a set of limiting assumptions. In using PN the missile guidance designer hopes to minimize required maneuvers, reduce consumption of power to the control surfaces, avoid the possibility of control surface saturation, and simultaneously reduce interception time and ZEM. Of course, minimizing control energy is not the only objective that may be posed. The pursuer may wish to instead minimize interception time, the ZEM, the total distance traveled, or any number of other objectives, depending on the specific scenario. In fact, given the rapid improvements in navigation technology, it may be unwise to simply assume a pursuing threat utilizes PN. But how can evasive strategies be proposed when the future actions of the pursuer are completely unknown? The answer lies in the theory of Differential Games.

2.4.1 Differential Games.

It has so far been assumed that the guidance law of the pursuer was known, thus enabling the search for an optimal evasion strategy. However, if the guidance law is not known, it is conservative to assume that both the evader and pursuer behave optimally. This concept is at the heart of the theory of Differential Games. Originating in the 1960s [52], Differential Game theory deals with the situation where multiple players are competing to achieve a payoff, or objective. The state of each player is

constrained by a set of differential equations. The goal in applying Differential Game theory is to find optimal strategies for the players, including boundaries where certain solutions exist (such as capture vs escape).

One subset of Differential games is formed by the Pursuit-Evasion games, in which a team of pursuers attempt to capture one or more evaders. While many recent studies have focused on games with more than two players [69–71], traditionally studies have focused on one pursuer and one evader.

It is often assumed that players have full information about the current state of their adversary, allowing them to anticipate the strategy of the other player. Thus each player will predict the decisions and future state of the other, and compensate accordingly. Players competing in this manner may generate control strategies whose payoff lies at an equilibrium, where neither player can improve their own payoff by altering their own strategy. These control strategies, along with their resulting trajectories and payoff, constitute an equilibrium solution. Assuming that both players are attempting to minimize their payoff, if a player fails to choose the equilibrium strategy, the resulting payoff is detrimentally increased for that player. If J_E is the objective of the evader, and J_P the objective of the pursuer, while $\mathbf{u}_E$ and $\mathbf{u}_P$ are their respective control strategies, this can be written [72]

$$\begin{aligned} J_E(\mathbf{u}_E^*, \mathbf{u}_P^*) &\leq J_E(\mathbf{u}_E, \mathbf{u}_P^*) \\ J_P(\mathbf{u}_E^*, \mathbf{u}_P^*) &\leq J_P(\mathbf{u}_E^*, \mathbf{u}_P). \end{aligned} \tag{2.34}$$

Here the * indicates that the control strategy is optimal. This is equivalent to writing [3]

$$\begin{aligned} J_E(\mathbf{u}_E^*, \mathbf{u}_P^*) &= \min_{\mathbf{u}_E} J_E(\mathbf{u}_E, \mathbf{u}_P^*) \\ J_P(\mathbf{u}_E^*, \mathbf{u}_P^*) &= \min_{\mathbf{u}_P} J_P(\mathbf{u}_E^*, \mathbf{u}_P). \end{aligned} \tag{2.35}$$

This type of equilibrium is often termed a Nash Equilibrium in the literature [73], although Nash's original paper only guarantees an equilibrium solution for games with mixed strategies [74]. A mixed strategy, unlike a pure strategy, is one where the decisions made by a player are probabilistic. The name Nash equilibrium does help to distinguish from other types of equilibrium, such as Pareto or Stackleberg [75]. There is in fact no guarantee that an equilibrium solution exists in general for a game where players use pure strategies [76].

In general, each player may have a separate objective, defined as

$$\begin{aligned} J_E &= \phi_E(\mathbf{x}_E(t_f), \mathbf{x}_P(t_f)) + \int_{t_0}^{t_f} L_E(t, \mathbf{x}_E, \mathbf{x}_P, \mathbf{u}_E, \mathbf{u}_P)dt \\ J_P &= \phi_P(\mathbf{x}_E(t_f), \mathbf{x}_P(t_f)) + \int_{t_0}^{t_f} L_P(t, \mathbf{x}_E, \mathbf{x}_P, \mathbf{u}_E, \mathbf{u}_P)dt. \end{aligned} \tag{2.36}$$

2.4.1.1 Zero-Sum Games.

In a two player game, if the objective of one player is the opposite sign of the objective of their adversary, the game is called zero-sum. Formulating the game in this way means that only a single objective needs to be written as

$$J_E(\mathbf{u}_E, \mathbf{u}_P) = -J_P(\mathbf{u}_E, \mathbf{u}_P) = J(\mathbf{u}_E, \mathbf{u}_P). \tag{2.37}$$

This zero-sum J has the same structure as the one sided objective from Equation (2.2),

$$J = \phi(\mathbf{x}_E(t_f), \mathbf{x}_P(t_f), t_f) + \int_{t_0}^{t_f} L(\mathbf{x}_E(t), \mathbf{x}_P(t), \mathbf{u}_E(t), \mathbf{u}_P(t), t)dt. \tag{2.38}$$

The pursuer seeks to minimize the objective of a zero-sum game, while the evader non-cooperatively seeks to maximize it. Isaacs names the minimax objective the

Value of the game, and it is written

$$V = \min_{\mathbf{u}_P} \max_{\mathbf{u}_E} J. \tag{2.39}$$

An example of this type of game would be for an evader to attempt to maximize the distance from the pursuer at some final time, while the pursuer minimizes this same distance.

Now the equilibrium relationship in Equation (2.34) can be rewritten as

$$J(\mathbf{u}_E, \mathbf{u}_P^*) \leq J(\mathbf{u}_E^*, \mathbf{u}_P^*) \leq J(\mathbf{u}_E^*, \mathbf{u}_P), \tag{2.40}$$

which restates that if either player fails to follow the equilibrium (now minimax) strategy, then the objective is worse for that player, either decreased or increased, respectively. However, it also implies that the player who did not change their strategy will improve their objective. In a zero-sum game, an improvement for one player comes at the expense of their adversary.

Often in pursuit-evasion problems, the dynamics and running cost are separable, meaning that

$$\begin{aligned} f(t, \mathbf{x}_E, \mathbf{x}_P, \mathbf{u}_E, \mathbf{u}_P) &= f_E(t, \mathbf{x}_E, \mathbf{u}_E) + f_P(t, \mathbf{x}_P, \mathbf{u}_P) \\ L(t, \mathbf{x}_E, \mathbf{x}_P, \mathbf{u}_E, \mathbf{u}_P) &= L_E(t, \mathbf{x}_E, \mathbf{u}_E) + L_P(t, \mathbf{x}_P, \mathbf{u}_P). \end{aligned} \tag{2.41}$$

In this case the Hamiltonian can be separately defined as

$$\begin{aligned} H_E(t, \mathbf{x}_E, \boldsymbol{\lambda}_E, \mathbf{u}_E) &= L_E(t, \mathbf{x}_E, \mathbf{u}_E) + \boldsymbol{\lambda}_E^T \mathbf{f}_E(t, \mathbf{x}_E, \mathbf{u}_E) \\ H_P(t, \mathbf{x}_P, \boldsymbol{\lambda}_P, \mathbf{u}_P) &= L_P(t, \mathbf{x}_P, \mathbf{u}_P) + \boldsymbol{\lambda}_P^T \mathbf{f}_P(t, \mathbf{x}_P, \mathbf{u}_P). \end{aligned} \tag{2.42}$$

The necessary optimality equations are then defined by the state equations [72]

$$\begin{aligned}\dot{\mathbf{x}}_E^*(t) &= \mathbf{f}_E(t, \mathbf{x}_E^*, \mathbf{u}_E^*) \\ \dot{\mathbf{x}}_P^*(t) &= \mathbf{f}_P(t, \mathbf{x}_P^*, \mathbf{u}_P^*),\end{aligned} \tag{2.43}$$

the adjoint equations,

$$\dot{\boldsymbol{\lambda}}_E^*(t) = -\frac{\partial H_E}{\partial \mathbf{x}_E}(t, \mathbf{x}_E^*, \boldsymbol{\lambda}_E^*, \mathbf{u}_E^*) \tag{2.44a}$$

$$\dot{\boldsymbol{\lambda}}_P^*(t) = -\frac{\partial H_P}{\partial \mathbf{x}_P}(t, \mathbf{x}_P^*, \boldsymbol{\lambda}_P^*, \mathbf{u}_P^*), \tag{2.44b}$$

and the stationarity conditions,

$$\mathbf{u}_E^*(t) = \underset{u_E}{\operatorname{argmin}}\, H_E(t, \boldsymbol{\lambda}_E, \mathbf{x}_E^*, \mathbf{u}_E) \tag{2.45a}$$

$$\mathbf{u}_P^*(t) = \underset{u_P}{\operatorname{argmin}}\, H_P(t, \boldsymbol{\lambda}_P, \mathbf{x}_P^*, \mathbf{u}_P). \tag{2.45b}$$

This set of necessary conditions can be compared to the one-sided optimal control necessary conditions in Equation (2.6). Typically a pursuit-evasion game has fixed initial conditions

$$\mathbf{x}^*(0) = \mathbf{x}_0, \tag{2.46}$$

but the terminal conditions depend on the specific problem. Two types of terminal conditions will be presented in this document. If the final time is fixed, but the final state is free, the terminal conditions are

$$\boldsymbol{\lambda}_E^*(t_f) = \frac{\partial}{\partial \mathbf{x}_E}\phi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) \tag{2.47a}$$

$$\boldsymbol{\lambda}_P^*(t_f) = \frac{\partial}{\partial \mathbf{x}_P}\phi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)), \tag{2.47b}$$

where above ϕ is the terminal cost associated with the final time, t_f, defined in Equation (2.2). Of the necessary optimality conditions defined in Equations (2.43)-(2.47), it is also the only function of both $\mathbf{x}_E$ and $\mathbf{x}_P$, serving to link the pursuer to the evader. All other equations are completely separate. These equations form the TPBVP which must be solved to find a minimax solution to a fixed terminal time zero-sum differential game.

If instead the final time is free, but the final state is constrained to a terminal surface, Ψ, the terminal conditions are

$$\Psi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) = 0 \tag{2.48a}$$

$$\boldsymbol{\lambda}_E^*(t_f) = \frac{\partial}{\partial \mathbf{x}_E}\phi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) - \nu\frac{\partial}{\partial \mathbf{x}_E}\Psi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) \tag{2.48b}$$

$$\boldsymbol{\lambda}_P^*(t_f) = \frac{\partial}{\partial \mathbf{x}_P}\phi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) - \nu\frac{\partial}{\partial \mathbf{x}_P}\Psi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) \tag{2.48c}$$

$$H_E(t, \mathbf{x}_E^*, \boldsymbol{\lambda}_E^*, \mathbf{u}_E^*) + H_P(t, \mathbf{x}_P^*, \boldsymbol{\lambda}_P^*, \mathbf{u}_P^*) + \frac{\partial}{\partial t}\phi(\mathbf{x}_E^*(t_f), \mathbf{x}_P^*(t_f)) = 0, \tag{2.48d}$$

where ν is a positive scalar Lagrange Multiplier. Now Equations (2.43) - (2.46) and (2.48) are the necessary optimality conditions, forming the slightly more complicated TPBVP which must be solved to find a minimax solution to the free final time zero-sum pursuit-evasion game. This has been done analytically for many simple problems; however, the solution of realistic problems with complex dynamics is typically accomplished computationally.

It is helpful to point out a few limitations with the necessary optimality conditions for a minimax solution [75]. First, it is not guaranteed that a minimax solution exists in general. If a solution is found, the necessary conditions must be satisfied, but they do not give any indication as to whether there is a solution to be found in the first place. Second, there may be more than one minimax solution, and they need not be unique.

2.4.2 Calculation of Minimax Solutions Using Collocation Methods.

Since the 1960s there have been numerous examples of pursuit-evasion solutions calculated by analytical (or geometrical) methods [52, 76–80]. Because of the difficulty in developing equilibrium solutions, most of these studies are devoted to dynamics within the horizontal plane, although some three dimensional analytical studies exist [81]. In order to capture the important relationship between energy and maneuverability, it is desirable to study the problem with increased fidelity. More recently, focus has been given to finding minimax solutions computationally. Unfortunately, direct collocation methods cannot be used immediately to calculate minimax solutions, primarily due to the NLP solvers which only minimize a single objective. Thus, in order to use established direct collocation techniques, it is necessary to adapt the method. While various algorithms have been proposed for computing minimax solutions for pursuit-evasion games [82–84], two are particularly promising for use in finding solutions to differential games via the PS method.

2.4.2.1 Semi-DCNLP Method.

One interesting method which is easily applied via collocation is known as semi-Direct Collocation Nonlinear Programming (semi-DCNLP) [85]. This method, derived from the work of [86], combines the indirect solution of solving the two-point boundary value problem associated with the necessary optimality conditions with the direct method of transcribing the discretized equations of motion to an NLP. This is done by defining the necessary optimality conditions for one player, and including them as state variables in the formulation of a one-sided optimal control problem, which is then solved using a direct method. By only indirectly solving for half of the necessary conditions, and directly solving the other half, the technique attempts to alleviate the difficulty of solving the two-point boundary value problem by adding

additional complexity to the setup of the direct problem. Several distinct differential game problems have been solved by this method, including fighter aircraft engaged in a dogfight [85], ballistic missile defense [87], and orbital pursuit-evasion [88].

Problems solved via the semi-DCNLP method may either directly solve for the control of the pursuer or the evader. These problems are labeled O_P and O_E, respectively. For a fixed final time, free final state problem, the general definition of problem O_P is: *Choose* $\mathbf{u}_P$ *to minimize the objective in Equation* (2.38) *subject to the dynamics in Equation* (2.43), *the costate dynamics in Equation* (2.44a), *the stationarity condition in Equation* (2.45a), *the initial conditions in Equation* (2.46), *and the terminal conditions in Equation* (2.47a).

For a fixed final time problem, O_E is defined: *Choose* $\mathbf{u}_E$ *to maximize the objective in Equation* (2.38) *subject to the dynamics in Equation* (2.43), *the costate dynamics in Equation* (2.44b), *the stationarity condition in Equation* (2.45b), *the initial conditions in Equation* (2.46), *and the terminal conditions in Equation* (2.47b).

If the problem has a free final time, the necessary conditions must additionally include the terminal surface and transversality condition. Thus O_P is defined: *Choose* $\mathbf{u}_P$ *to minimize the objective in Equation* (2.38) *subject to the dynamics in Equation* (2.43), *the costate dynamics in Equation* (2.44a), *the stationarity condition in Equation* (2.45a), *the initial conditions in Equation* (2.46), *the terminal conditions in Equations* (2.48a) *and* (2.48b), *and the transversality condition Equation* (2.48d).

Finally, the free final time problem O_E is defined: *Choose* $\mathbf{u}_E$ *to minimize the objective in Equation* (2.38) *subject to the dynamics in Equation* (2.43), *the costate dynamics in Equation* (2.44b), *the stationarity condition in Equation* (2.45b), *the initial conditions in Equation* (2.46), *the terminal conditions in Equations* (2.48a) *and* (2.48c), *and the transversality condition Equation* (2.48d).

Curiously, although free final time problems O_E and O_P require additional terminal conditions, in references [85], [87], and [88] several of these are not enforced. This is because they sometimes require the terminal costates from both players, while only one is available. However, it is possible that neglecting terminal conditions would result in a solution which is not strictly a minimax. In [89], it is suggested that both O_E and O_P must be solved, and the resulting objective compared, in order to ensure that a solution represents a minimax. This is not a problem with fixed final time problems with no terminal surface, because the transversality condition does not apply, and the final costates are only functions of the states.

Another difficulty with this method is in providing an initial guess of the costates for the indirect player. In [90] and [91] a genetic algorithm is used to generate the initial guess. In another recent study [92] it is indicated that this genetic algorithm pre-processor requires a large amount of computation time, and suggests a sensitivity-homotopy method as an alternative.

2.4.2.2 Iterative Methods.

One early approach for obtaining equilibrium solutions, called the "cycling method", involved repeated solution of two one-sided optimal control problems [93]. This process appears to have been independently discovered later by others, and termed "Iterative Relaxation" [3, 94, 95]. This method was specifically developed for computing solutions rapidly and in a stable manner. Stability is defined as the ability of the method to converge to an equilibrium solution regardless of the initial guess. Since the algorithm involves solving one-sided problems for each player, it is also desirable that convergence does not depend on the order in which one-sided solutions are obtained. This property is termed "asynchronous", and is particularly useful for distributed computing of the solutions.

The iterative technique is intuitively simple. First, a one-sided optimization problem for the evader is posed by assuming an initial suboptimal policy for the pursuer. The control solution to this one-sided problem is optimal with regards to the pursuer's suboptimal policy. The evader's optimal control solution is then used as the suboptimal policy in a one-sided optimal control problem in favor of the pursuer. This produces an updated control policy for the pursuer, which can in turn be used against the evader, etc. This can more succinctly be written [3], assuming that each player desires to minimize their respective objective,

$$\begin{aligned} \mathbf{u}_{E,k+1} &= \underset{\mathbf{u}_E}{\operatorname{argmin}}\, J_E(\mathbf{u}_E, \mathbf{u}_{P,k}) \\ \mathbf{u}_{P,k+1} &= \underset{\mathbf{u}_P}{\operatorname{argmin}}\, J_P(\mathbf{u}_{E,k+1}, \mathbf{u}_P). \end{aligned} \tag{2.49}$$

This describes a process where the players take turns responding to updated control information from their adversary. If the process were only accomplished for a single iteration, the solution trajectories and controls would represent a Stackleberg equilibrium [75]. However, after repeated application the solution may converge to a stable Nash-type equilibrium solution. In order to allow memory of the past policies of each player, a relaxation parameter, $0 \leq \alpha \leq 1$ is used. The algorithm becomes

$$\begin{aligned} \mathbf{u}_{E,k+1} &= \alpha_E \mathbf{u}_{E,k} + (1-\alpha_E) \underset{\mathbf{u}_E}{\operatorname{argmin}}\, J_E(\mathbf{u}_E, u_{P,k}) \\ \mathbf{u}_{P,k+1} &= \alpha_P \mathbf{u}_{P,k} + (1-\alpha_P) \underset{\mathbf{u}_P}{\operatorname{argmin}}\, J_P(\mathbf{u}_{E,k+1}, \mathbf{u}_P). \end{aligned} \tag{2.50}$$

This relaxation parameter tends to reduce large jumps in each solution, as only a portion of the newly found one-sided control is applied at each iteration. One significant problem with this method is the lack of guarantee of converging to a solution. In the references, conditions for existence and stability of equilibria for static games are proposed; however, the extension to dynamic games, especially as

discretized into a NLP, is not clear. Regardless, variations on this method have been used to solve complex pursuit-evasion type problems [96, 97].

One particular variety of iterative technique, termed the Decomposition Method [59, 83, 98], appears promising. As described above, separate evader and pursuer one sided optimal control problems are solved iteratively. However, the control strategies and objective values are not shared between iterations. Instead, the pursuer is given the objective, J_P, of minimizing the final time to reach the evader's last known position, given as

$$\mathbf{e} = (x_E(t_f), y_E(t_f), h_E(t_f))^T. \tag{2.51}$$

where x_E, y_E, and h_E are the east, north, and altitude coordinates of the evader. The actual final position of the evader is

$$\mathbf{z}_E = (x_E(t_f), y_E(t_f), h_E(t_f))^T, \tag{2.52}$$

and the actual final position of the pursuer is

$$\mathbf{z}_P = (x_P(t_f), y_P(t_f), h_P(t_f))^T. \tag{2.53}$$

The capture condition the pursuer hopes to achieve is then

$$\boldsymbol{\psi} = (\mathbf{e} - \mathbf{z}_P)^T = \mathbf{0}, \tag{2.54}$$

and the objective is the final time. The value of the pursuer is the minimized objective

$$V_P = \min_{\mathbf{u}_P} t_f, \tag{2.55}$$

subject to the capture condition, initial conditions, and equations of motion. This forms a free final time, fixed final state one-sided optimal control problem which only involves the pursuer's equations of motion and the final position of the evader.

In response, the evader wishes to maximize the pursuer's value. This is done by first linearizing V_P about $\mathbf{e}$ as

$$V_P \approx \tilde{V}_P(\mathbf{e}, \mathbf{z}_P) + \left(\frac{\partial V_P}{\partial \mathbf{e}}\right)^T (\mathbf{z}_E - \mathbf{e}) \tag{2.56}$$

where the gradient of V_P with respect to $\mathbf{e}$ is given by

$$\frac{\partial V_P}{\partial \mathbf{e}} = \frac{\partial}{\partial \mathbf{e}}\phi(\mathbf{e}, \mathbf{z}_P) + \mathbf{b}^T \frac{\partial}{\partial \mathbf{e}}\boldsymbol{\psi}(\mathbf{e}, \mathbf{z}_P). \tag{2.57}$$

Above, $\mathbf{b}$ is the Lagrange multiplier vector associated with the capture condition, $\boldsymbol{\psi} = 0$. These multipliers represent the sensitivity of the objective to the constraints, and will serve to guide the evader in responding to the pursuer's optimal trajectory. The evader's objective is to maximize the pursuer's value at the final time, which can be accomplished by setting

$$V_E = \max_{\mathbf{u}_E} \left(\frac{\partial V_P}{\partial \mathbf{e}}\right)^T (\mathbf{z}_E - \mathbf{e}). \tag{2.58}$$

This forms a one-sided, fixed final time free final state optimal control problem for the evader which requires only the terminal position $\mathbf{e}$ and Lagrange multipliers from the previously solved pursuer problem. The resulting trajectory is used to set a new value for $\mathbf{e}$, to which the pursuer will then respond.

The algorithm is relatively simple once the two problems have been defined. The steps are:

1. Generate an initial guess for $\mathbf{e}^i$, with $i = 0$.

2. Minimize the time for the pursuer to reach $\mathbf{e}^i$, record the Lagrange multipliers $\mathbf{b}$.

3. Maximize the evader's objective given by Equations (2.58) and (2.57), record the new value of $\mathbf{e}^{i+1}$.

4. Check $\epsilon = |\mathbf{e}^{i+1} - \mathbf{e}^i|$. If ϵ is small, stop. Otherwise, return to Step 2.

While the algorithm requires repeated solution of two separate one-sided optimal control problems, it is very easy to implement. It avoids forming the TPBVP representing the necessary optimality conditions, instead obtaining the solution directly. It may therefore solve problems with inequality or path constraints which would be difficult for a method such as semi-DCNLP or Indirect Transcription.

2.4.3 Real Time Optimal Control.

One method recently used at the Air Force Institute of Technology (AFIT) for dealing with uncertainty when solving optimal control problems is to repeatedly solve a deterministic problem, each time modifying the initial conditions of the problem to match real-time observations. The optimal control problem is only solved for a finite time horizon, which moves along as the problem progresses in real time. This method captures uncertainty simply by admitting that variations from the optimal trajectory will occur, and then adjusting the trajectory with a new optimal solution. There is some ambiguity about the name of this method. When optimization is performed for a fixed final time problem the method will here be termed Receding Horizon Control (RHC). If the optimal control problem is free final time, it will be referred to as Real Time Optimal Control(RTOC). These methods are sometimes viewed as a feedback controller with coarse time steps [99], although in the presence of nonzero-mean or

time-correlated disturbances this statement is overly optimistic [100]. The method is also known as Model Predictive Control [101–103].

Recently at AFIT an RTOC controller was implemented in a study which sought to control the motion of a quadrotor UAV to land on a wire [9, 104]. In the study, the author proposed a dual control problem, where the objective of the optimal control problem had two parts, a primary mission to control the UAV and a secondary mission to estimate its location with respect to the wire. The estimation requirement was embedded into the constraints rather than the objective, and the optimal control problem was solved in real-time using a PS method. Uncertainty from measurements were included in the optimal control problem via the Unscented Kalman Filter [105–109].

Another technique for propagating uncertainty in optimal control was demonstrated for the automatic air collision avoidance problem [10, 110]. Unlike the ground collision avoidance problem, in air collision scenarios an intruding object moves through the intended flight path of an aircraft with an unknown future trajectory. The aircraft is required to both sense the obstacle, and then take action to avoid it. In order to estimate the future path of the intruding aircraft, a Particle Filter (PF) was implemented [111]. In the PF estimation of the intruding aircraft, the uncertain nature of the future path is modeled by a number of particles sampled from an assumed distribution of the initial state of the intruder. Each particle is propagated forward up to a specified time horizon using standard aircraft equations of motion. The spread of particles forms a distribution from which statistics of the flight path, such as mean position and standard deviation, may be calculated. Rather than use the statistics, however, the author chose to enclose all the particles inside a convex hull representing the region where a collision would occur. An ellipsoid was fit to this hull at each time step, and an interpolation technique was developed to describe

the changing ellipsoid through time. An optimal control problem was then posed wherein the evading aircraft was required to maneuver around the region enclosed in the ellipsoids.

It was then assumed that measurements of the intruder would be available to the evading aircraft at certain time intervals. The estimated location of the particles was updated according to a measurement model, the particles were propagated, a convex hull constructed, fit to ellipsoids, and interpolated. An optimal control problem was then solved to keep the aircraft from entering the interpolated ellipsoid region at a given time. This process was implemented as an RHC problem, allowing the aircraft to continually adjust its trajectory as measurements were obtained of the intruding aircraft. Unfortunately the PF required propagation of 10,000 particles, which could not be performed in real-time on a desktop computer during simulation. It is likely that this limitation could be overcome in flight by processing the propagation step of the algorithm in parallel using specialized hardware. One benefit to using a PF is that it can be used to model nonlinear dynamics, and is not limited to the Gaussian assumption.

Finally, RHC was used at AFIT with the PS method to solve a ground collision avoidance problem, where two different approaches were compared to help a low flying heavy aircraft avoid colliding with terrain [11, 112]. The first approach maximized the distance the aircraft approached the terrain during a pull-up maneuver, while constraining the controls. The second approach minimized the control usage, while constraining the distance within which the aircraft approached the ground. The deterministic optimal control problems were solved serially as the simulated aircraft approached a ground obstacle, and were implemented at the last moment the RHC indicated that evasion was still possible, including some amount of buffer for safety. Although real-time calculation of these trajectories was not achieved in practice, it

was demonstrated that the minimum control formulation tended to have a faster computation speed. This approach, while not directly propagating random variables in time, captures the stochastic nature of flight by recalculating optimal trajectories as conditions change over time.

The success of these studies suggest that either RHC or RTOC could be a useful tool in dealing with uncertainty in a Pursuit-Evasion problem. While efforts will be made to reduce optimal control computation time to fit real-time requirements, the main goal of the current research is to compensate for the uncertainty and demonstrate the feasibility of the solution methods.

2.5 Summary

Optimal control is ideal for studying pursuit-evasion, because it represents the problem as finding a control to minimize an objective subject to dynamic and static constraints. The choice of objective and constraints is important, as they define the problem and thus the solution. In the one-sided optimal control pursuit-evasion problem, a suboptimal behavior must be assumed for either the pursuer or evader, and here a set of common suboptimal pursuit and evasion strategies have been described. A differential game removes the suboptimal behavior by allowing that both the pursuer and evader may act optimally, given assumptions on the information available to each. Two techniques have been described for how a pursuit-evasion game may be solved using computational techniques. Finally, several methods have been described for the scenario when full information about an adversary is not available at all times, or the equations which model their motion are subject to uncertainty. While this chapter has been an overview of methods for solving these problems, the next chapter will present the specific models and algorithms to be used in the current research.

III. Methodology

3.1 Overview

In order to explore the role of energy in aircraft-missile pursuit-evasion scenarios, it is important that the dynamics of the aircraft and the missile realistically represent the current energy state of each vehicle. This means that the equations of motion must capture the exchange of potential energy and kinetic energy through altitude and velocity. Additionally, thrust and drag must accurately generate and dissipate energy. Thus the models of the aircraft and missile will be presented here, along with details highlighting their energy exchange characteristics.

Several atmospheric properties must be defined for both models. It is assumed that gravity is a constant, g_0=9.8066 m/s^2. The temperature and density of the atmosphere, in K and kg/m^3 is dependent on altitude in meters by the relations [113]

$$\begin{aligned} T &= T_0 - ah \\ \rho &= \rho_0 \left(\frac{T_0 - ah}{T_0} \right)^{n-1} \end{aligned} \tag{3.1}$$

where standard density at sea level, ρ_0, is 1.225 kg/m^3, standard temperature, T_0, is 288.16 K, and the lapse rate, a, is 0.0065 K/m. The value of the dimensionless constant n is 5.2561. These formulas are valid up to altitudes at the edge of the tropopause, or 11 km. For subsonic and supersonic flows, the speed of sound is a function of temperature alone,

$$s = \sqrt{\gamma RT}. \tag{3.2}$$

3.2 Models

3.2.1 Evader Model.

In order to properly capture the exchange of kinetic and potential energy of an aircraft during a missile evasion encounter, the equations of motion must include all three dimensions, as well as viscous drag and thrust terms. However, a full six degree of freedom set of equations would be overly complicated and burdensome for the optimal control solver. For this reason, a point-mass model is adopted which assumes that the aircraft is able to maintain stability within given operating limits. Thus modeling the control surface deflections is avoided, along with the need to track the exact pitch, roll, and yaw angles. Instead it is assumed that the pilot or autopilot is able to command a desired angle of attack (α) and bank angle (μ). It will be assumed that the engagements take place over a relatively short distance such that the curvature and rotation of the Earth are negligible. Throttle will not be modeled in this work, as all the problems solved in this work simply require maximum thrust. This is intuitive from an energy generation perspective. Also, it will be assumed that due to the short duration of the encounter, the mass of the evading aircraft, m_E, is constant at 19,051 kg. Its surface area, s_E, used for lift and drag calculations, is 49 m^2 [65].

Eight state variables are required to describe the motion of the aircraft in the velocity and flat-Earth frames: easting (x_E), northing (y_E), altitude (h_E), velocity (V_E), flight path angle (γ_E), heading angle (χ_E), angle of attack (α_E), and bank angle (μ_E). The dynamics can be represented by a system of first-order coupled differential equations [44].

$$
\begin{aligned}
\dot{x}_E &= V_E \cos\gamma_E \cos\chi_E \\
\dot{y}_E &= V_E \cos\gamma_E \sin\chi_E \\
\dot{h}_E &= V_E \sin\gamma_E \\
\dot{V}_E &= \frac{1}{m_E}(T_E \cos\alpha_E - D_E) - g\sin\gamma_E \\
\dot{\gamma}_E &= \frac{1}{m_E V_E}((T_E \sin\alpha_E + L_E)\cos\mu_E - g\ m_E \cos\gamma_E) \\
\dot{\chi}_E &= \frac{\sin\mu_E}{m_E V_E \cos\gamma_E}(T_E \sin\alpha_E + L_E) \\
\dot{\alpha}_E &= u_\alpha \\
\dot{\mu}_E &= u_\mu
\end{aligned}
\tag{3.3}
$$

In order to avoid stall, the evader is limited to angles of attack between -5 and 15 degrees. A structural limit of 9 g is also placed on the aircraft. To smooth the solutions obtained numerically via the PS method, it is helpful to model the angle of attack and bank angle as additional states in order to reduce numerical chatter. Thus, the controls are set to be the angle of attack rate and the bank angle rate, which are limited as

$$
\begin{aligned}
-\dot{\alpha}_M \leq \dot{\alpha}_E \leq \dot{\alpha}_M \\
-\dot{\mu}_M \leq \dot{\mu}_E \leq \dot{\mu}_M,
\end{aligned}
\tag{3.4}
$$

where $\dot{\alpha}_M$ is 15 degrees per second and miu_M is 180 degrees per second.

The lift (L_E), drag (D_E), and maximum available thrust (T_E) are functions of altitude, Mach number, and angle of attack. These are modeled after the tabulated data of a supersonic interceptor aircraft found in [65]. For the semi-DCNLP method, it is necessary to calculate partial derivatives of the lift, drag, and thrust with respect to altitude, velocity, and angle of attack. Numerical approximations using the coarse

tabulated data were found to be inaccurate, therefore piecewise functions were fit to the data such that the derivatives were continuous.

3.2.1.1 Evader Lift, Drag, and Thrust Approximations.

The lift force is calculated by the classic expression

$$L = qS_E C_{L,\alpha} \alpha_E \tag{3.5}$$

where the dynamic pressure, q, is defined as $q = 0.5\rho V_E^2$. The lift coefficient, $C_{L,\alpha}$ varies with Mach number, and the tabulated data is fit by

$$C_{L,\alpha} = \begin{cases} 3.44 + e^{-200(M-1)^2} & M \leq 1.121 \\ 4.12 - 1.8\sqrt{M-1} & M > 1.121 \end{cases}. \tag{3.6}$$

The lift coefficient and its derivative have been plotted versus Mach number in Figure 2 to show that the fit is only an approximation, but that the derivative is continuous.

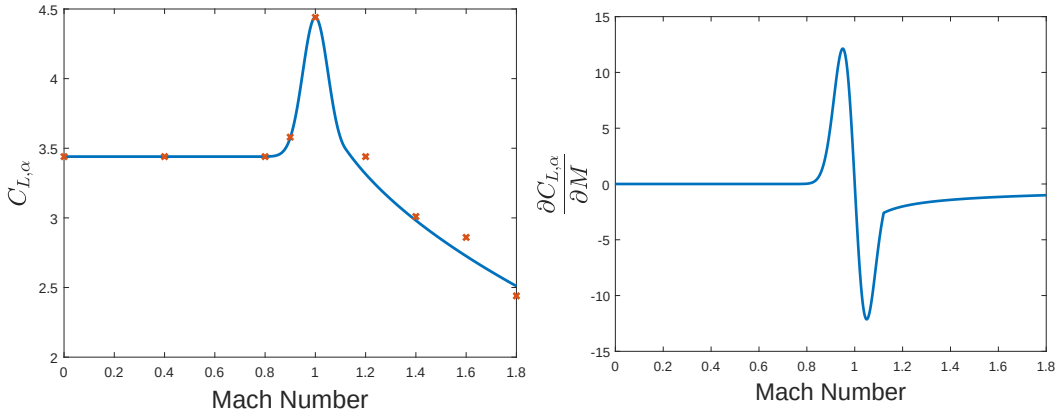

Figure 2. Approximate fit for the lift coefficient, $C_{L,\alpha}$.

Drag force is calculated using the zero lift drag coefficient, $C_{D,0}$ and the factor η which relates the drag to the lift coefficient and angle of attack squared.

$$D = qS_E(C_{D,0} + \eta C_{L,\alpha}\alpha^2) \tag{3.7}$$

Both $C_{D,0}$ and η are functions of Mach number. The approximation for $C_{D,0}$ is

$$C_{D,0} = \begin{cases} 0.013 + 0.03e^{-80(M-1.1)^2} & M \leq 1.104 \\ 0.04748 - 0.014\sqrt{M-1} & M > 1.104 \end{cases}, \tag{3.8}$$

which has been plotted versus Mach number in Figure 3.

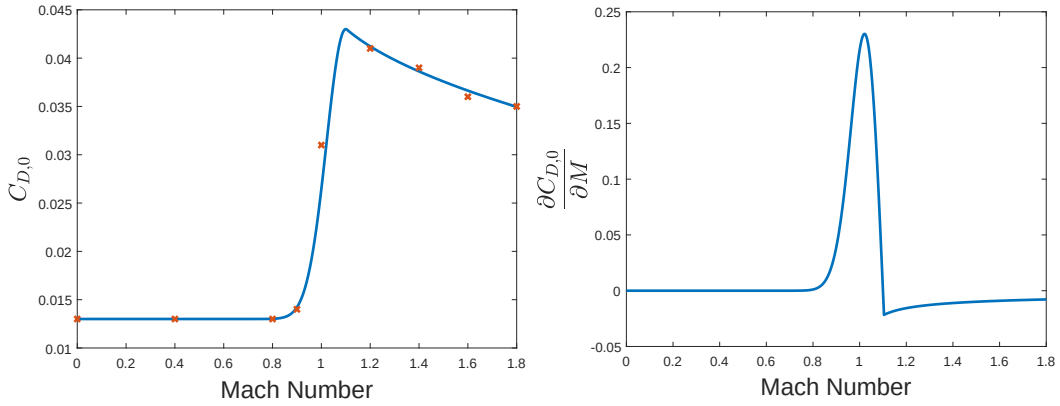

Figure 3. Approximate fit for the zero lift drag coefficient, $C_{D,0}$.

Finally, the factor η is given by

$$\eta = 0.54 + \frac{0.39}{1 + e^{-10(M-1)}} \tag{3.9}$$

and is pictured in Figure 4.

The maximum available thrust is a function of Mach number and velocity. A polynomial fit was found using linear regression for altitude in meters to be

$$T_E = 106790 + 35323M - 8.0766h_E + 25752M^2 - 3.6352Mh_E + 0.000177h_E^2, \tag{3.10}$$

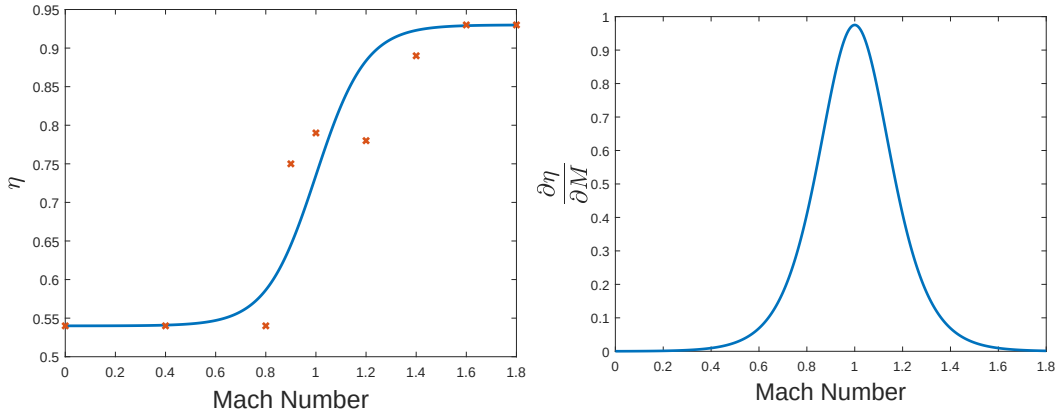

Figure 4. Approximate fit for η.

and is plotted in Figure 5.

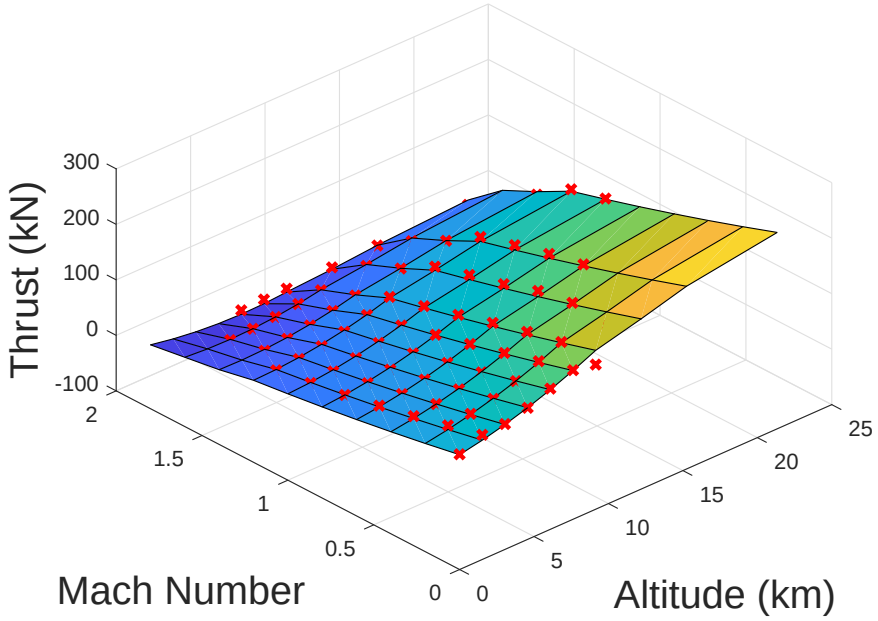

Figure 5. Approximate fit for T_E.

The maneuverability of the aircraft can be described by the "doghouse" plot and the V-n diagram, which display maximum turn rate (rad/s) and maximum load factor (g) at a given velocity and altitude condition. Figure 6 displays these two charts for a variety of altitudes. Perhaps the most interesting feature of the two diagrams is the movement of the corner velocity, which occurs where the stall limit meets the max g limit. While the corner velocity increases for increasing altitude, the actual

turn rate or g available decreases. Clearly, to improve maneuverability, the aircraft must descend to the lowest altitude possible, and accelerate to the corner velocity or higher. There is a large bump in both the turn rate and the max g available near Mach 1.0, and in fact for some altitudes this is the point of highest turn rate.

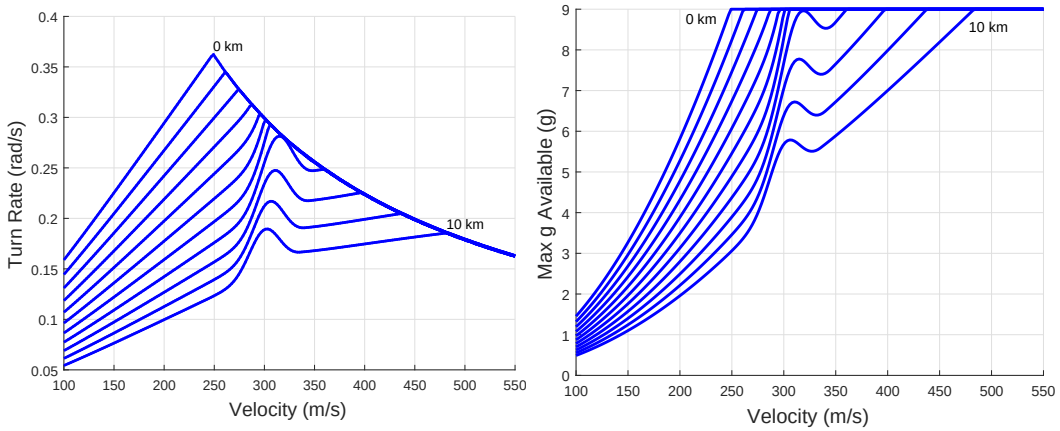

Figure 6. Turn rate and available g versus velocity for a variety of altitudes.

To conserve energy, the aircraft should fly at altitude and velocity combinations where the specific power is greater than zero. However, in order to achieve high maneuverability, the aircraft will need to cross into regions where specific power is less than zero, meaning that it will trade energy for maneuverability. The turn rate and available g plots have been recreated in Figure 7 for an altitude of 8 km, but this time the zero specific power line has been included to demarcate the energy and maneuverability zones.

3.2.2 Pursuer Model.

The pursuer is also modeled in three dimensions, on a flat non-rotating Earth. Rather than model angle of attack and bank, it is assumed that the missile commands lateral and longitudinal accelerations, $a_{\gamma,c}$ and $a_{\chi,c}$. The navigation system is subject to lag, thus the actual accelerations relate to the commanded values by first-order dynamics by the time constant τ.

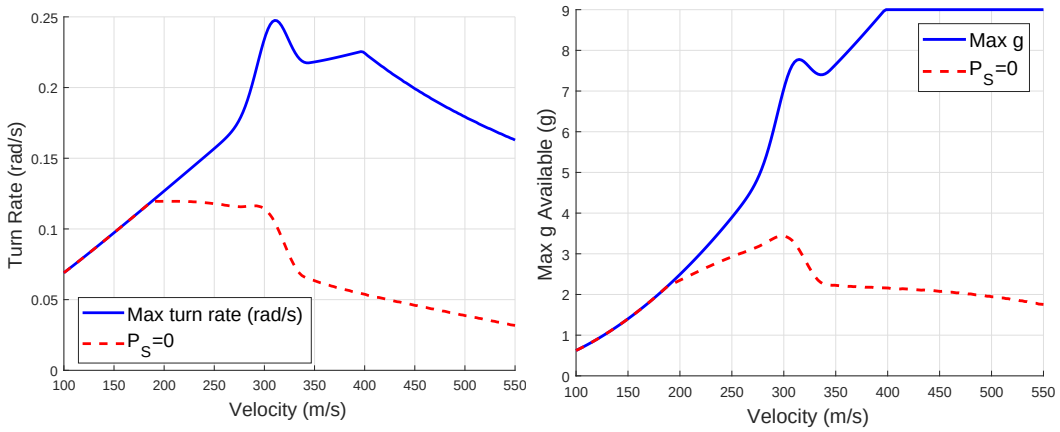

Figure 7. P_S=0 line versus velocity at 8 km altitude.

The dynamics are given by the equations

$$
\begin{aligned}
\dot{x}_P &= V_P \cos\gamma_P \cos\chi_P \\
\dot{y}_P &= V_P \cos\gamma_P \sin\chi_P \\
\dot{h}_P &= V_P \sin\gamma_P \\
\dot{V_P} &= \frac{1}{m_P(t)}(T_P(t) - D_P) - g\sin\gamma_P \\
\dot{\gamma_P} &= \frac{1}{V_P}(a_\gamma - g\cos\gamma_P) \\
\dot{\chi_P} &= \frac{a_\chi}{V_P \cos\gamma_P} \\
\dot{a}_\gamma &= \frac{a_{\gamma,c} - a_\gamma}{\tau} \\
\dot{a}_\chi &= \frac{a_{\chi,c} - a_\chi}{\tau}.
\end{aligned}
\tag{3.11}
$$

Figure 8 shows the geometry of the engagement, specifically the relationship between the flight path and heading angles to the x, y, and h directions, along with the angle of attack and LOS.

The drag on the missile varies with the square of the magnitude of the applied acceleration, which is the two-norm of the lateral and normal accelerations,

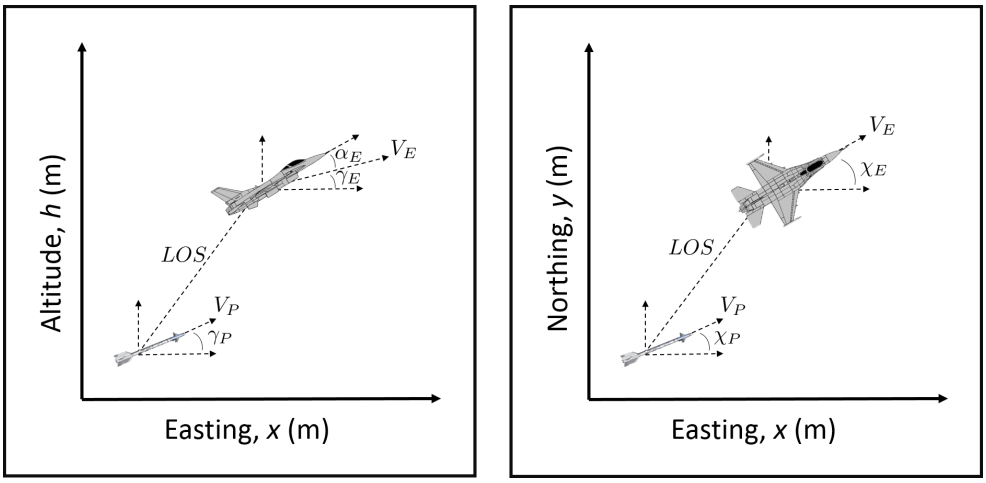

Figure 8. A visual depiction of the engagement geometry.

$$a_D = \sqrt{a_\gamma^2 + a_\chi^2}. \tag{3.12}$$

The drag is then calculated as [114]

$$D_P = \frac{1}{2}\rho S_P V_P^2 C_{D,0} + \frac{2km_P^2}{\rho S_P}\left(\frac{a_D{}^2}{V_P}\right). \tag{3.13}$$

The engine produces a maximum thrust of T_P for t_P seconds, and the mass is depleted as propellant is consumed. Parameters for the missile have been modified from reference [114] to increase the range and max velocity. They are displayed in Table 3.

Table 3. Table of modeling parameters for the missile.

m (kg)	S_P (m^2)	$C_{D,0}$	k	g-limit	τ (s)	T_P (N)	t_P (s)	ISP (s)
173.6	0.0324	0.1	0.03	35	0.5	20000	8	250

The g limit is enforced on the magnitude of the acceleration vector, rather than on the separate components of acceleration. This is called the circular or isotropic

vectogram [56], where saturation occurs in both guidance channels simultaneously, representing a missile which is ambivalent to roll orientation.

The concept of the corner velocity does not exist for the current missile, because aerodynamic stall has not been included in the model. This means that the max turn rate is described completely by the 35 g limit. At the missile's peak velocity near 1500 m/s, the turn rate is approximately the same as that of the aircraft at its corner velocity. However, at lower velocities the missile's max turn rate increases inversely to the decrease in velocity. In the limit as the missile velocity approaches zero, the turn rate becomes infinite. This is a problem with the fidelity of the model, but these low velocities are never reached in scenarios within this work.

Perhaps more important for the missile during the endgame phase is the time constant, τ. This system lag causes the missile's actual acceleration to lag behind the command. The effect of this lag can be seen in Figure 9, where a missile which responds immediately to guidance acceleration commands is compared to a missile with a first-order lag, as given by Equation (3.11). The missile with a non-zero time constant cannot achieve a given cross-range value as quickly as the missile with no lag, meaning that effectively the missile's turning radius in increased. In the endgame maneuver, the evader must pass within the pursuer's minimum turn radius to achieve a non-zero final miss distance, and the time constant significantly deteriorates the pursuer's ability to maneuver to prevent this from occurring.

There is a frequency aspect to the terminal evasion maneuver which is linked to the missile's time constant. If the evader can cause the pursuer's commands to fall out of phase with the actual accelerations by performing a "weave" type maneuver, it can achieve a relatively large final miss distance [28]. One such maneuver is the High g Barrel Roll (HGBR), which is performed by pulling maximum g while simultaneously rolling at a constant rate [61], which causes the longitudinal and lateral guidance

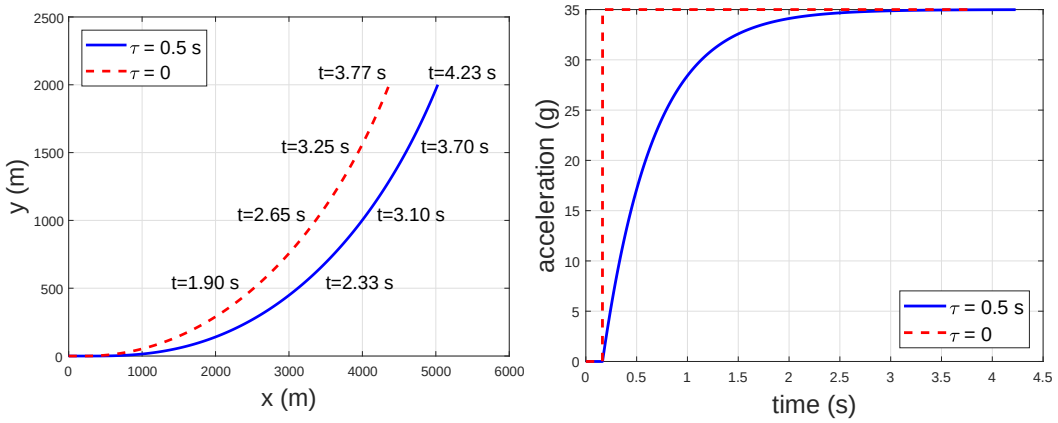

Figure 9. Effect of the time constant on the missile's turning performance. Time is shown at y distance intervals of 500 meters.

channels commands of the missile to oscillate. The specific roll rate and maneuver start time are critical to the performance of the HGBR, and optimal values depend on the guidance parameters, such as the time constant and the navigation gain N_P for a missile using PN. The longitudinal and lateral accelerations induced in a PN missile chasing an aircraft performing a HGBR are shown in Figure 10, distinctly out of phase.

To show the sensitivity of the HGBR with respect to roll rate, a number of maneuvers have been simulated against a PN guided missile with an initial separation of 5 km. The orientation of the engagement is a tail chase, beginning at the same altitude. It is assumed that the missile has already expended its thrust. Each HGBR is applied at the maneuver start time by increasing the angle of attack to 15 degrees, while rolling at a fixed rate. The maneuver start time is adjusted from 0 to nearly 5 seconds, and the final miss distance, or the moment when the closing velocity reaches zero, was recorded. This was done for a variety of roll rates between zero and 180 degrees per second, with zero representing a pull in the horizontal plane. The results, shown in Figure 11, indicate that there is an ideal roll rate and starting time for the maneuver, which depends on the value of τ and N_P. Note that for the scenario shown in Figure 11, with N_P=4 and τ=0.5 seconds, the ideal roll rate is approximately 2

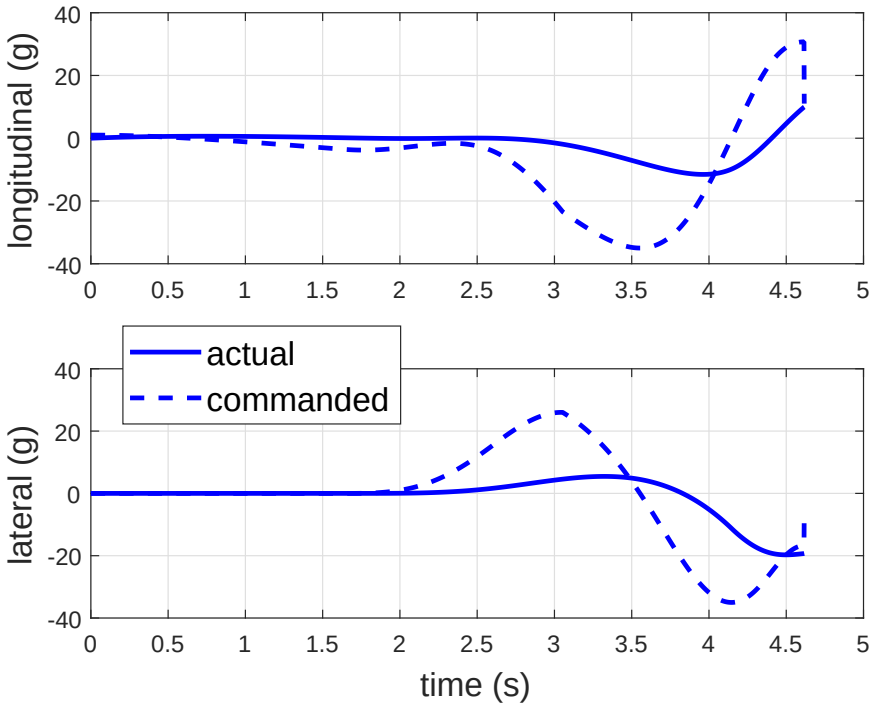

Figure 10. Missile longitudinal and lateral accelerations induced by a HGBR maneuver.

radians per second, while the maneuver start time corresponds to a time-to-go of 3 seconds, or nearly enough time to perform 2π radians of total roll. This does not necessarily hold true for other values of N_P and τ.

3.3 Summary

In order to accurately capture the physics involved in a missile-aircraft pursuit-evasion scenario, the equations of motion for each vehicle have been constructed to represent the exchange of energy between kinetic and potential, along with the generation and dissipation of energy due to thrust and drag. Diagrams showing the max turn rate and g versus velocity show the maneuverability of the aircraft at a variety of altitudes. The P_S=0 line on these diagrams indicates the conditions for which the evader will either gain or lose energy. In the endgame maneuver, the evader seeks to exploit the missile's guidance system in order to achieve a high final

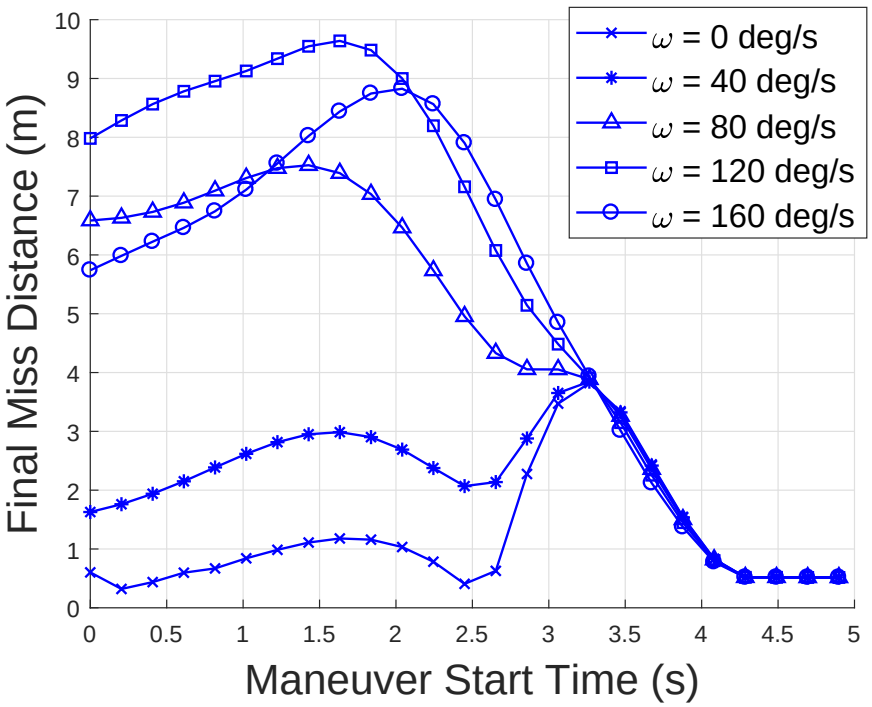

Figure 11. Final miss distance versus maneuver start time for a variety of HGBR roll rates.

miss distance. The HGBR maneuver is a good suboptimal choice for evasion which puts the missile's commanded and actual longitudinal and lateral accelerations out of phase. In the next chapter, optimal evasive trajectories will be presented which strongly resemble the HGBR during the terminal maneuver phase.

IV. One Sided Optimal Missile Evasion

The HGBR maneuver was shown to achieve relatively high final miss distance by causing oscillations in the missile's lateral and longitudinal acceleration channels. However, the miss distance depended on maneuver start time and roll rate, highlighting the time sensitive nature of final evasive maneuvers. While the HGBR is a suboptimal maneuver, its sensitivity to time indicates that an optimal maneuver may exist which will achieve the maximum miss distance at the moment of the missile's closest approach. The purpose of this chapter is to define and solve the optimal control problem which generates an open-loop control trajectory which, for given initial conditions, will result in the best chance of evading a single missile fired from medium range. While the open-loop control is only directly applicable to the specific initial conditions and modeling parameters posed in the optimal control problem, and thus not directly useful for a wide variety of scenarios, it can serve as a benchmark for comparison with other evasion maneuvers. Also, the optimal trajectory can be deconstructed to understand the dynamics of an evasion maneuver, and these lessons can be applied to a wider range of scenarios.

4.1 The Closest Point of Approach Problem

The main objective in optimal evasion is of course survival of the evader, represented by the inverse of the Probability of Kill, P_K. Minimizing this number may be accomplished in a variety of different ways (see the literature review), but the most direct way is to maximize the distance between the missile and the aircraft at the CPA. Thus the objective for the CPA problem is

$$J = -r(t_f). \tag{4.1}$$

It is often useful to add a small term to the objective to dampen control energy by including an integral function of control in the objective, which will reduce numerical chatter in the solution, along with other beneficial effects in describing the TPBVP which will be detailed later. Thus the objective is augmented to be

$$J = -r(t_f) + w_E \int_{t_0}^{t_f} \left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M}\right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M}\right)^2 dt, \tag{4.2}$$

where $\dot{\alpha}_M$ and $\dot{\mu}_M$ are scaling factors, set to the maximum angle of attack or bank rate. The weight, w_E, is chosen carefully to minimize the effect of the integral term on the final miss distance, usually such that the integral term is three orders of magnitude smaller.

A necessary condition for the CPA is that the first derivative of the range must be zero. The closing velocity, defined by Equation (2.17), is the negative of the range rate. This conditions defines the final time, and represents a transversal surface upon which the trajectory terminates,

$$V_C(t_f) = 0. \tag{4.3}$$

Thus the CPA problem can be classified as free final time, fixed final state.

In order to simplify the CPA problem, it is assumed that the problem begins after the missile's thrust has been expended. This removes the need to divide the problem into two phases in order to capture the change in discontinuous physics when the engine is shut off. At this moment the velocity of the missile is much higher than the aircraft, it has gained some altitude from its launch point due to lofting, and its mass has been reduced by the amount of expelled propellant. Although the missile is still climbing slightly at this time, the guidance is commanding a pitch down maneuver in order to descend to the altitude of the aircraft.

The initial angle of attack of the evader is set by trimming the aircraft to cruise at the given altitude and velocity. This sets the angle of attack and throttle. However, it is assumed that immediately upon detecting the missile, the throttle is set to full, while the angle of attack remains at the previous trim condition.

As an example scenario, the pursuing missile is launched toward the evading aircraft with an initial LOS (from the pursuer to the evader) of 35 degrees, from an initial range of 30 km, with an initial velocity of 290 m/s. However, during the eight second boost phase the missile quickly reaches a higher velocity and covers approximately 6 km. The initial conditions for both the evader and pursuer just after the missile's thrust ends are given in Table 4.

Table 4. Table of initial conditions for the evader and pursuer.

Evader

x_E (m)	y_E (m)	h_E (m)	V_E (m/s)	γ_E (deg)	χ_E (deg)	α_E (deg)	μ_E (deg)
20,054	14,042	10,000	290.2	0	0	2.91	0

Pursuer

x_P (m)	y_P (m)	h_P (m)	V_P (m/s)	γ_P (deg)	χ_P (deg)	a_γ (m/s^2)	a_χ (m/s^2)
0	0	10,573	1,424	5.33	35	-134	0

The CPA problem, defined by the objective from Equation (4.2), the terminal condition in Equation (4.3), the dynamic constraints given by Equations (3.3) and (3.11), the additional constraints on angle of attack, angle of attack rate, bank angle rate, and maximum g defined in Chapter III, and the initial conditions shown in Table 4, was solved using the GPOPS-II software. Although the PS method is in general capable of obtaining solutions from even poor initial guesses, for the current problem this proved to be somewhat difficult for reasons that will be explained shortly. A relatively good initial guess is required to solve this 16 state, 2 control, free final time problem.

4.1.1 Solution to the Closest Point of Approach Problem.

In order to obtain a solution, it was necessary to first solve the problem at a close range, 2 km, with an initial guess defined by forward integration of the equations of motion using a constant control. Then, the resolved optimal solution was saved as a new initial guess and the range was increased by a small amount. The problem was solved for the new range, and the new solution was again saved as an initial guess. The range was incremented and the problem re-solved in this homotopic manner until the actual desired initial range was achieved.

Because the PS method is based on gradients, it is common to find locally optimal solutions, and the homotopy method of producing the initial guess tends to aggravate this problem. Thus, once a full solution was achieved, the initial guess was perturbed and the problem was re-solved to search for other locally optimal trajectories. The initial guess was replaced with the highest objective trajectory found, until no further improvements could be made. The final solution was solved to an NLP tolerance of 1×10^{-6} and a mesh tolerance of 1×10^{-4}, using the Patterson mesh error from [22]. The final miss distance achieved by the optimal trajectory was 48.6 meters, significantly better than the HGBR results from Chapter III. A visualization of this trajectory is presented in Figures 12 - 17. As seen in Figure 12, the evader initially

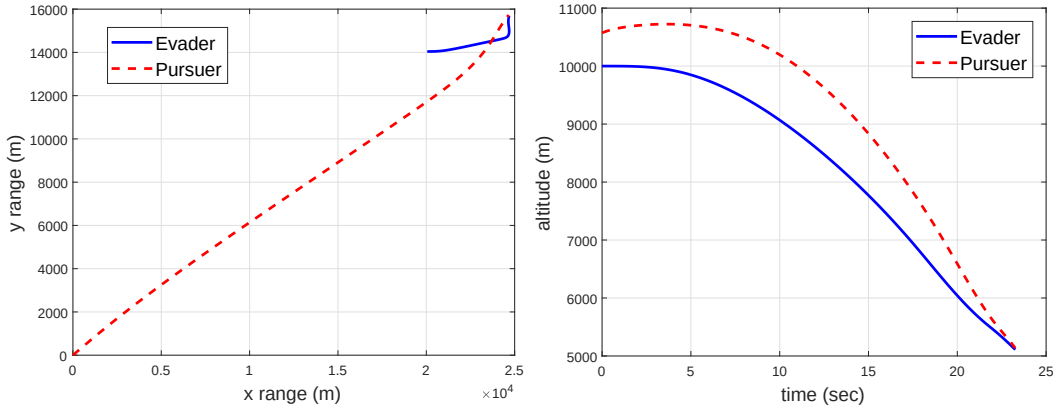

Figure 12. Overhead view (left) and altitude profile (right) of the CPA problem.

turns away from the pursuer while diving into the atmosphere. This is done by quickly rolling inverted and increasing the angle of attack smoothly so that a steep dive is obtained, trading altitude for speed, as seen in Figure 13. At approximately

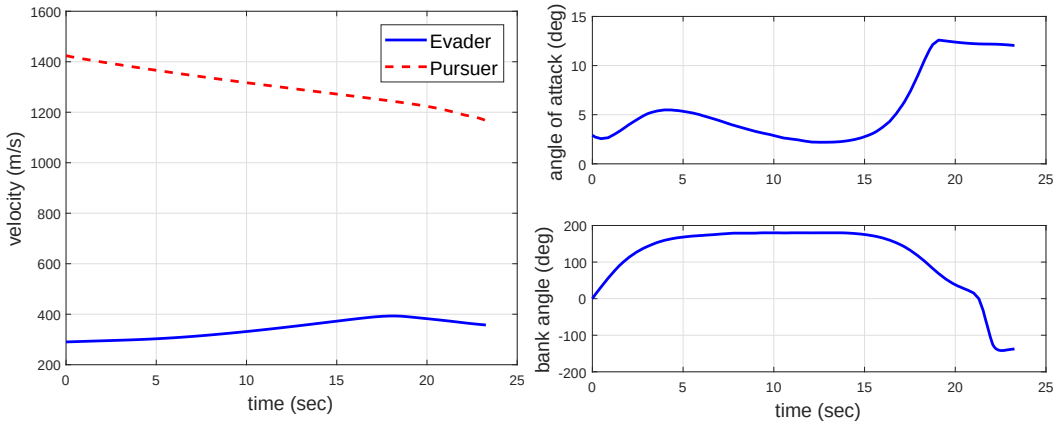

Figure 13. Velocity (left), angle of attack / bank angle (right) for the CPA problem.

t_{go} of 5 seconds the evader begins to roll in the opposite direction while greatly increasing the angle of attack. At this highly negative flight path angle, the heading rate change is large due to the cosine term in the denominator of the heading rate in Equation (3.3). Thus the aircraft changes direction quickly, as seen in Figure 14. This rapid change in both flight path angle and heading, caused by rolling while

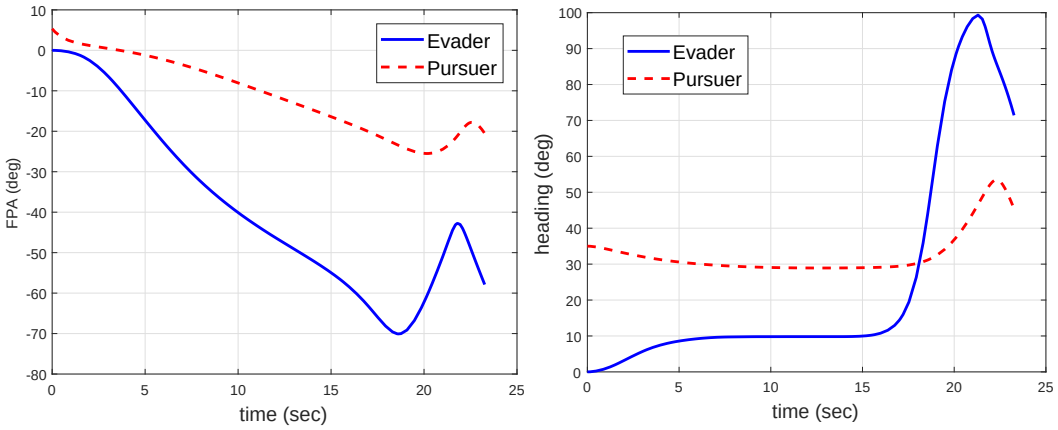

Figure 14. Flight Path Angle (left) and heading angle (right) for the CPA problem.

pulling maximum g in a dive, causes the PN controller on the missile to demand

oscillating accelerations in both longitudinal and lateral channels. Because of the time constant, the actual accelerations lag behind the commands, which saturate near the end of the engagement, as seen in Figure 15. This trajectory displays

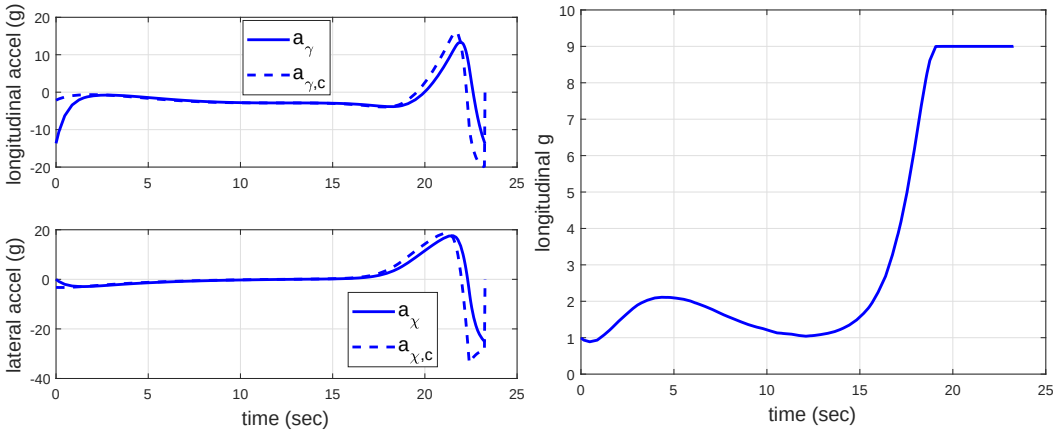

Figure 15. Missile accelerations (left) and aircraft g (right) for the CPA problem.

essentially two phases. First, an energy conserving phase where altitude is traded for velocity, positioning the aircraft at a steep flight path angle to enable a rapid change in heading. This is followed by a second phase where the evader applies large control effort to outmaneuver the pursuer by exploiting its navigation time constant and acceleration saturation. The first phase is dominated by energy management, while the second phase trades energy for maneuverability, as seen in Figure 16, which shows the specific power profile of both the evader and pursuer. During the first 15 seconds of the flight the evader maintains a positive P_S. The point where P_S crosses from positive to negative marks the beginning of the maneuverability phase. Another useful visualization is created by plotting the load factor, or longitudinal acceleration, of the aircraft versus velocity on top of a V-n diagram. This must be done in pieces, since the maximum g limits depend on the altitude, which changes during the trajectory. To capture the changes in altitude, three plots have been made for various times and altitudes along the trajectory. The maximum load factor and P_S=0 line have also been included as shown in Figure 17. One notable feature of

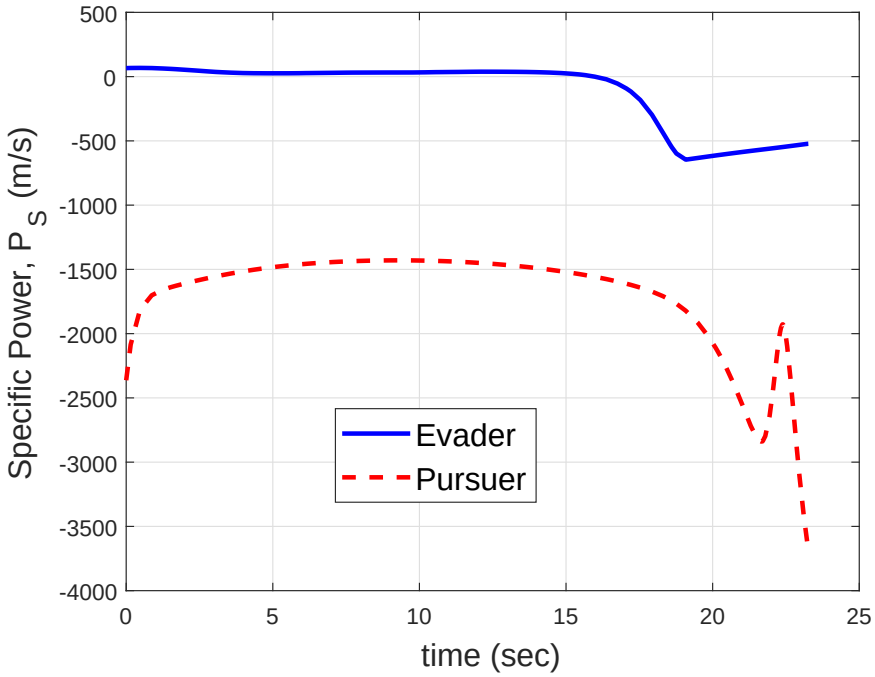

Figure 16. Specific power, P_S, for the CPA problem.

the trajectory is that during the dive, shown on the left image, the aircraft increases its load factor into a slight bump in the P_S=0 line near Mach 1. By exploiting this region, the aircraft is able to save energy while pulling more g's. Also, as seen in the center image, the aircraft enters the maneuverability phase and quickly pulls the maximum g of 9. This happens just above the corner velocity, meaning that it is able to pull maximum g and achieve the maximum turn rate. Finally, because the aircraft continues to descend, the corner velocity decreases, allowing the aircraft to continue to slow down while still achieving a high g pull (right image).

Interestingly, the maneuverability phase of the trajectory appears qualitatively similar to the HGBR maneuver as described in Chapter III. In both the maneuvers the evader pulls a high g load while rolling, although the exact roll timing is different between the two, and because the CPA is at lower altitude, it reaches a higher g load. While the HGBR applies a fixed roll rate, the optimal trajectory employs a variable roll rate to achieve a higher final miss distance. This indicates that, like

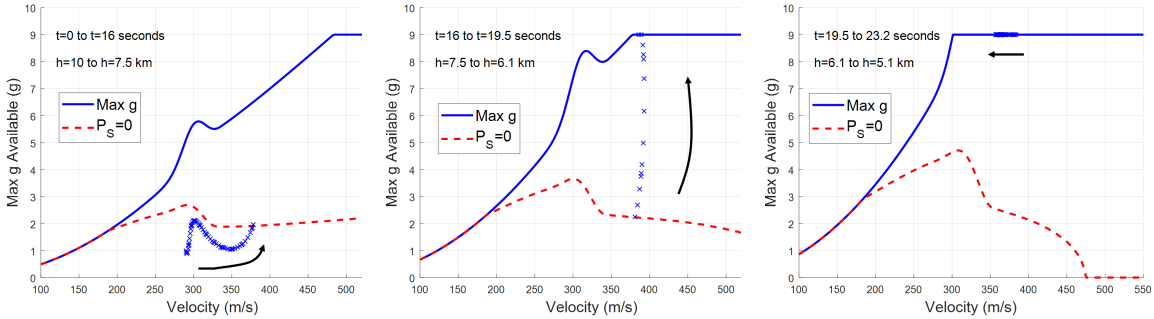

Figure 17. VN diagram showing load factor at three altitudes along the trajectory of the CPA problem.

the HGBR, the optimal maneuver must be precisely timed, an obvious difficulty in a realistic scenario where the precise state of the missile is uncertain. It also hints that the HGBR is nearly optimal, and may serve as a simplified, easily implemented replacement if an optimal result is unavailable.

4.1.2 Difficulties with the Closest Point of Approach Problem.

As mentioned, the CPA problem required a good initial guess to converge, and additional work to search for the globally optimal solution. The primary reason for this lies with the termination surface, V_C=0, which defines the end to the problem. Figure 18 shows the closing velocity for the CPA problem. Most notably, the closing velocity drops precipitously near the point of closest approach. Although the trajectory in Figure 18 ends at V_C=0, the closing velocity would continue to drop to a large negative value if the simulation were continued.

This jump in V_C does not always occur. For initial conditions where the evader is able to outrun the pursuer, the closing velocity will decrease steadily due to drag on the missile until it reaches zero. However, for these initial conditions the solution is somewhat trivial, as no evasive maneuver is necessary. For a missile fired within its operating range, the closing velocity will jump quickly to zero at the moment of closest approach. This sudden change in V_C is problematic for a solver such as SNOPT,

which uses gradient calculations to explore the solution space, causing difficulties in convergence primarily due to infeasibilities related to the terminal constraint. Once a feasible solution has been obtained, the mesh must also be refined multiple times to reduce discretization error associated with the large change in V_C.

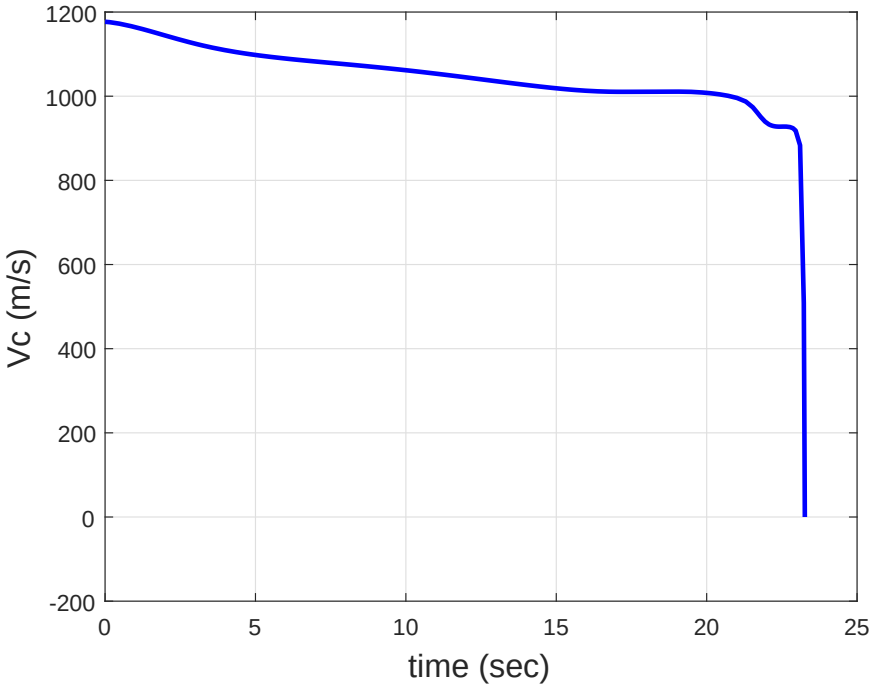

Figure 18. Closing velocity, V_C, for the CPA problem.

In the end, the problem can be solved using the methods described, but depending on the initial guess, the solution may require an unacceptably long computation time. Therefore it makes sense to re-pose the optimal control problem. As has been demonstrated here, there are two distinct phases in the optimal trajectory, an energy management phase and a maneuverability phase. The approach for the rest of this work is to divide the optimal evasion problem into these two phases. The maneuverability phase will continue to be posed as the CPA problem. However, the energy management phase will be posed with a different objective, and both fixed and free final time versions of this problem will be explored.

4.2 The Fixed Final Time Problem

In the fixed final time version of the energy management problem, it is assumed that the problem termination is defined by the final time alone. The most obvious objective at this final time is to maximize the distance between the pursuer and evader, as in

$$J = -r(t_f) + w_E \int_{t_0}^{t_f} \left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M}\right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M}\right)^2 dt. \tag{4.4}$$

Note that the form of the objective has not changed from Equation (4.2), except that the terminal constraint from Equation (4.3) has been removed. This means the final state is unconstrained, resulting in a much easier problem to solve. The value of the weight, w_E, must be re-tuned for the larger value of $r(t_f)$. The fixed final time optimal control problem is defined by the objective in Equation (4.4), a predetermined final time, the dynamic constraints given by Equations (3.3) and (3.11), the additional constraints on angle of attack, angle of attack rate, bank angle rate, and maximum g defined in Chapter III, and the initial conditions shown in Table 4. This will be known in short as the Fixed Time (FX) problem, and it was solved with the GPOPS-II software using an initial guess found by propagating the equations of motion via a Runge-Kutta four step (RK4) integration with zero control inputs. All solutions were obtained with an NLP tolerance of 1×10^{-6} and a mesh tolerance of 1×10^{-4}. Figure 19 shows the overhead view and altitude profile of a solution calculated with t_f=18 seconds.

At the beginning of the trajectory, the evader's control effort is largely focused on turning away from the pursuer. However, this is done with a positive specific power, as seen in Figure 20, meaning that the evader is attempting to conserve energy. Despite this qualitative similarity between the energy management phase of the FX and the CPA trajectory, closer comparison of Figures 19 and 12 shows some differences. For

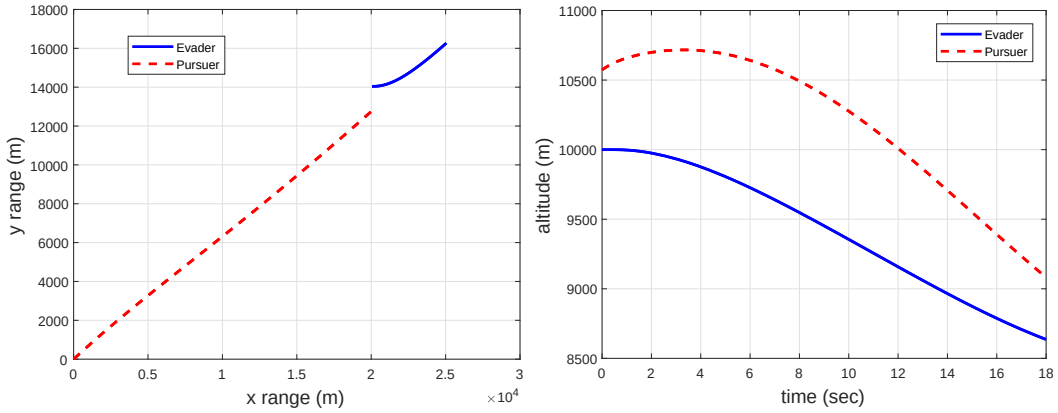

Figure 19. Overhead view (left) and altitude profile (right) of the Fixed Time problem.

example, the CPA trajectory dives much more quickly, reaching nearly 2 km lower in altitude by 18 seconds. Also, while the CPA trajectory rolls completely inverted, the FX only rolls to about 120 degrees, then slowly rolls back toward zero bank. This is because the CPA "knows" it must prepare for the final evasive maneuver, whereas this information is missing from the FX problem.

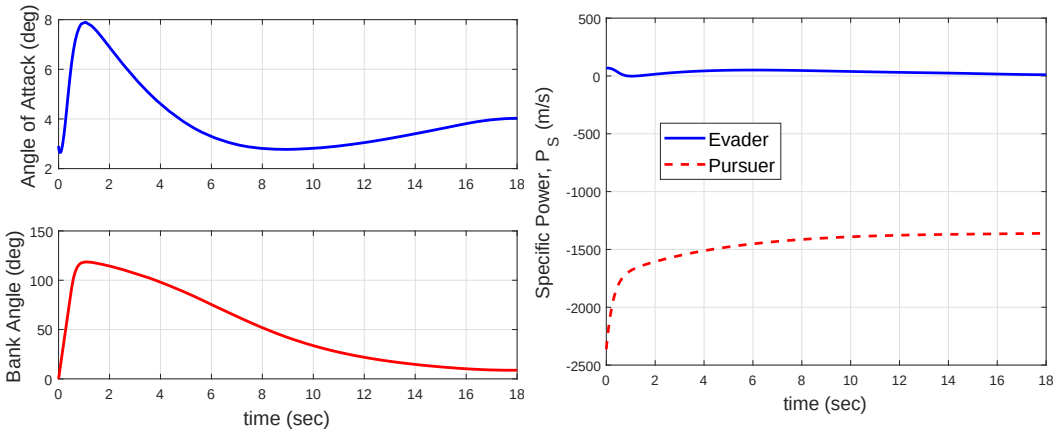

Figure 20. Angle of attack / bank angle (left) and P_S (right) of the Fixed Time problem.

The control weight term, w_E, represents a balance between the scalar and running costs in Equation (4.4). It is important that the weight does not adversely affect the scalar objective. To verify the correct weight, several problems were solved with varying weights. The scalar objective was recorded, along with the computation time required to find a solution. Error was calculated based on the lowest weight value,

10^{-2}. The magnitude of the error percentage and the calculation time are displayed in Figure 21, showing that the error increases with the weight. Interestingly, the computation time required actually increases for the FX problem for large weights. The optimal weight for the computation time, near w_E=2, gives an error of less than 0.01 percent.

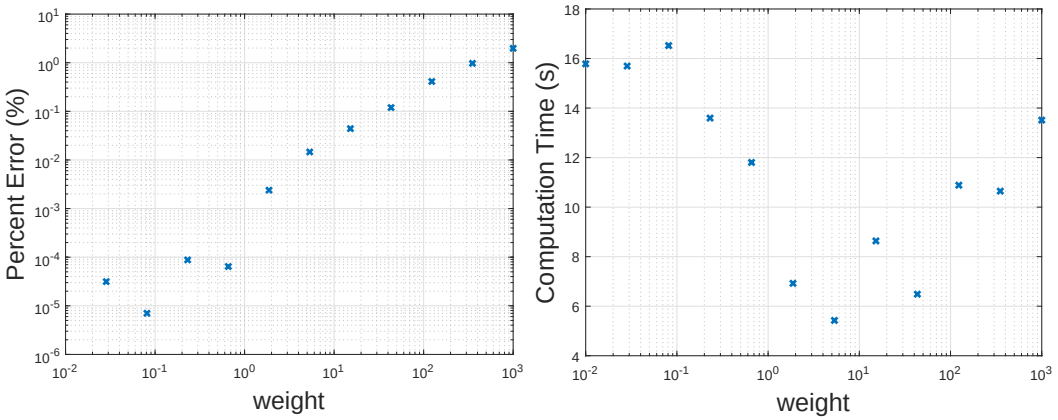

Figure 21. Magnitude percentage error (left) and computation time (right) versus control energy weight for the FX problem.

Posing the FX problem requires an assumption of the game of kind, namely that the pursuer will not intercept the evader before the final time. Clearly, the final time for this problem should be set prior to the beginning of the maneuverability phase. Unfortunately, unless the CPA problem has already been solved, there is no clear distinction when this phase should begin. To gain an understanding of how to set t_f, a series of FX problems were solved with identical initial conditions but varying terminal times. The final conditions of each FX problem were then used as the initial conditions of the CPA problem, creating a hybrid FX / CPA problem. The final miss distance from the hybrid problem was recorded and plotted versus the varying fixed final time, here called the break time, in Figure 22. Also included are the results of simply cruising until the break time, and then applying the CPA problem. The hybrid cruise - CPA results are meant to serve as a baseline; what would happen if no action were taken during the energy management phase.

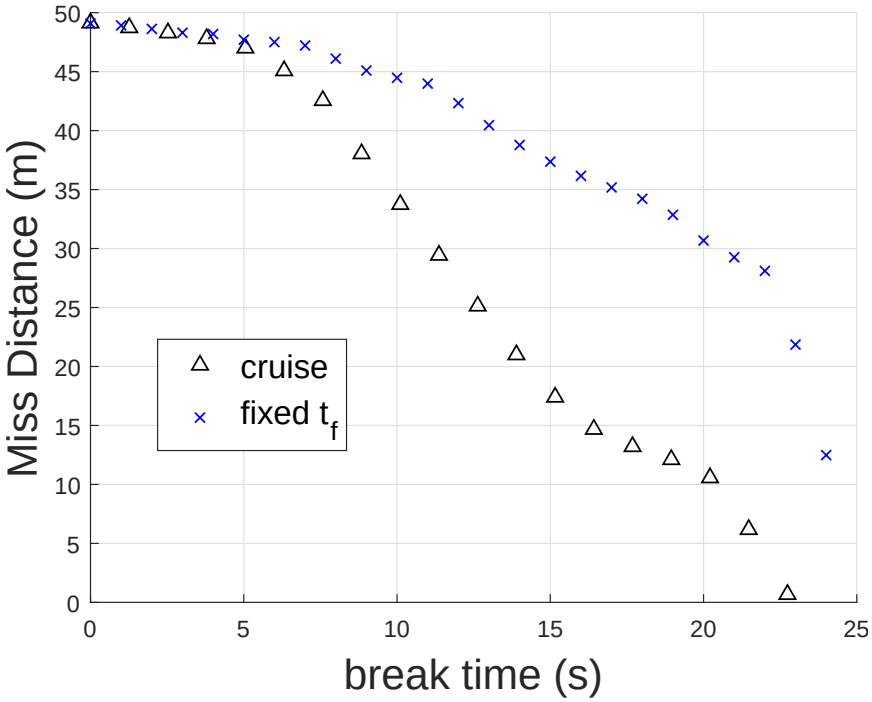

Figure 22. Results of varying the final time, t_f, for the FX problem.

On the left side of Figure 22, where the break time is zero seconds, the miss distance is equivalent to that achieved by the CPA problem. However, as the break time between the FX and CPA problems increases, the miss distance decreases. This is because the FX problem does not prepare for the final evasive maneuver as well as the CPA problem. For the example trajectory shown previously where the final time was set to 18 seconds, the resulting final miss distance from the hybrid would be approximately 34 meters, meaning a loss of 15 meters from the pure CPA solution. However, the time to calculate the solution has been reduced dramatically. Even with a good initial guess, the full CPA solution starting from the initial conditions requires approximately an order of magnitude longer to calculate than the combined FX - CPA problem. No quantified comparison of the computation times between the pure CPA problem and the hybrid FX - CPA problem has been made because the solution convergence depends too strongly on the initial guess, which may be different

for every scenario. In general however, the hybrid problem requires significantly less time to calculate.

4.3 The Free Final Time Problem

In the free final time version of the energy management problem it is assumed that the pursuer will intercept the evader at some unknown time. Thus the objective for the evader is to delay this time as much as possible, giving the objective function

$$J = -t_f + w_E \int_{t_0}^{t_f} \left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M}\right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M}\right)^2 dt, \tag{4.5}$$

where the final time t_f is defined by the intercept condition, $r(t_f) = 0$. Once again, the control energy weight term, w_E, must be tuned such that the effect of the control damping on the scalar value of the objective is insignificant.

The free final time optimal control problem is defined by the objective in Equation (4.5), the condition $r(t_f) = 0$, the dynamic constraints given by Equations (3.3) and (3.11), the additional constraints on angle of attack, angle of attack rate, bank angle rate, and maximum g defined in Chapter III, and the initial conditions shown in Table 4. This is the Free Time (FR) problem, and similar to the FX problem, it was solved with the GPOPS-II software using an RK4 propagated initial guess. As with the FX problem, the problem was solved in relatively short computation time, despite the fact that like the CPA problem it is has a free final time and a constrained terminal surface. Unlike the CPA problem, however, the terminal surface does not have a large gradient. In fact, the variable which describes the terminal surface for the FR problem, r, is the integral of the variable for the terminal surface of the CPA problem, V_C, and benefits from the smoothing action of the integration.

The overhead view and altitude profile are shown in Figure 23, while the angle of attack, bank angle, and P_S are shown in Figure 24. Although the FR and FX

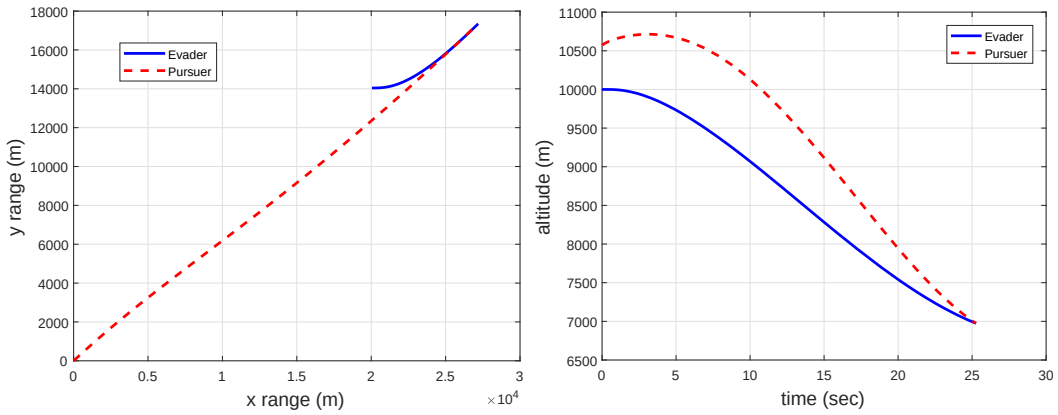

Figure 23. Overhead view (left) and altitude profile (right) of the FR problem.

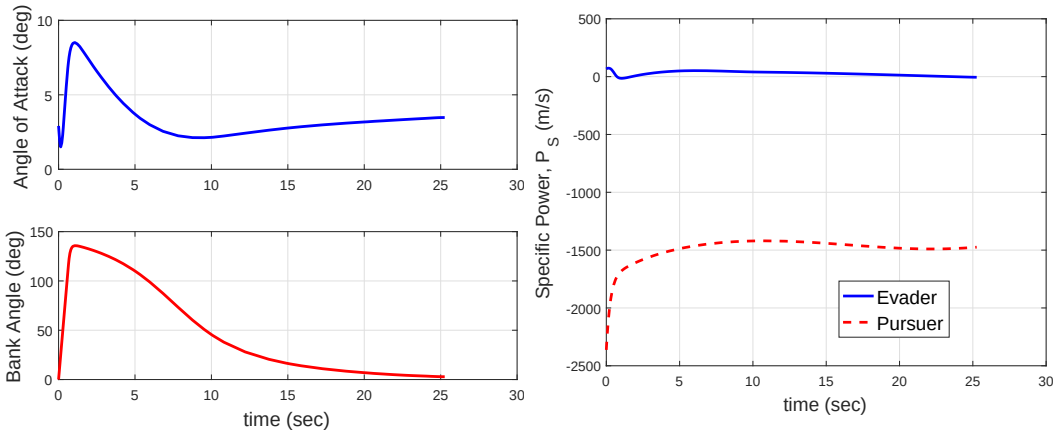

Figure 24. Angle of attack / bank angle (left) and P_S (right) of the FR problem.

trajectories seem qualitatively similar, they differ quantitatively in an important way. The FR problem dives deeper into the atmosphere at any given time. This is because the FX problem is "short sighted", meaning that it makes no attempt to project required performance beyond the fixed final time at 18 seconds. This causes the evader to turn more sharply in order to achieve the same heading as the pursuer in a shorter time. In the FR trajectory, the evader places less priority on the turn in order to descend further into the atmosphere where it has a drag advantage over the pursuer. This can be seen by comparing the P_S profiles in Figures 20 and 24. In both

trajectories the P_S profile dips during the turn, then after recovering, slowly descends until the final time. However, the P_S for the FR trajectory descends more slowly. At t =18 s the value of P_S is approximately 15 m/s for the FR trajectory, while it is 10 m/s for the FX. This shows that the FR trajectory conserves energy slightly better than the FX.

Once again the appropriate value of w_E must be demonstrated. The weight was varied and the value of the final time was recorded. Figure 25 shows the magnitude of the percentage error of the final time, and the calculation time required to solve the problem for a variety of weight values. A value of $w_E = 0.2$ minimizes computation time and keeps the error below 10^{-2}.

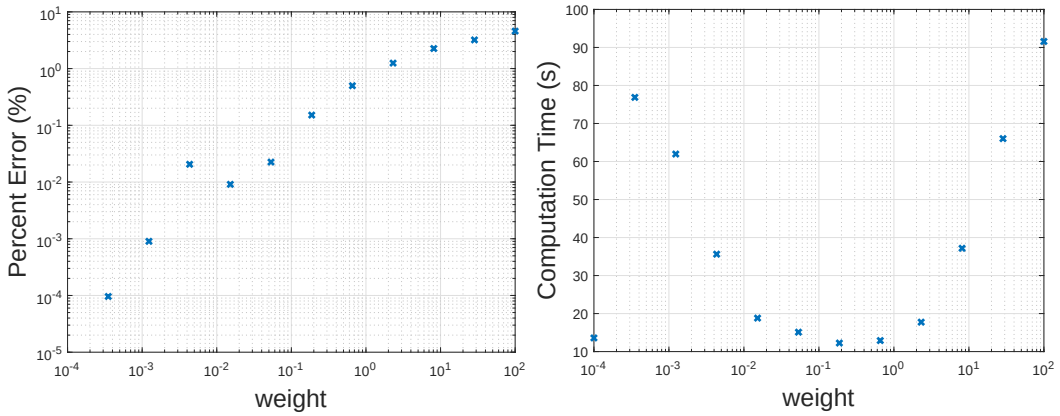

Figure 25. Magnitude percentage error (left) and computation time (right) versus control energy weight for the FR problem.

The difference between the FX and FR problems can be seen by varying the final time in the FX problem, and comparing the trajectories directly. Various FX trajectories have been plotted with the FR trajectory from the same initial conditions in Figure 26. As t_f for the FX problem approaches the final value for the FR problem, the FX trajectory converges to the FR trajectory. Each of the FX trajectories with a smaller value for t_f lacks information about times beyond t_f, and thus "greedily" maximizes $r(t_f)$ without considering the future performance. Solving the FR problem takes the entire time horizon into account.

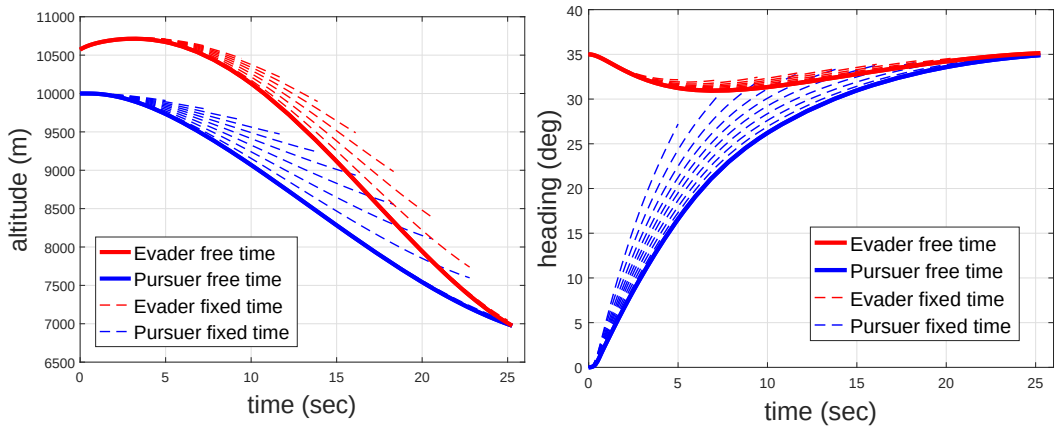

Figure 26. Altitude (left) and heading (right) of the FX trajectory for various final times compared to the FR trajectory.

The benefit of solving the FR problem can be directly seen by solving the hybrid FR / CPA problem for various break times, as was done for the hybrid FX / CPA problem. The resulting final miss distance was plotted versus the break time in Figure 27, along with the results from cruising and the FX trajectory originally seen in Figure 22. Clearly, the FR trajectory serves as a better substitute to the energy management phase of the CPA problem than the FX problem. The final miss distance performance of the CPA problem is nearly achieved during the first 17 seconds of flight time, corresponding to t_{go} of approximately 7 seconds. Therefore, if the evader flies the FR trajectory until $t_{go} = 7$ seconds, then implements an evasion maneuver by solving the CPA problem, it will still achieve nearly the highest possible miss distance.

Figure 27 demonstrates that the FR problem, when used as an energy management phase coupled with a maneuverability phase, achieves a higher miss distance than the FX problem, but only for this one scenario. This effect can be demonstrated more generally by checking the results at different engagement geometries. This has been done by varying the initial bearing between the pursuer and evader from 0 through 180 degrees, keeping the initial range constant. The break time for the FR solution was set to be $t_{go} = 5$ seconds for each scenario. This same value was used as the fixed

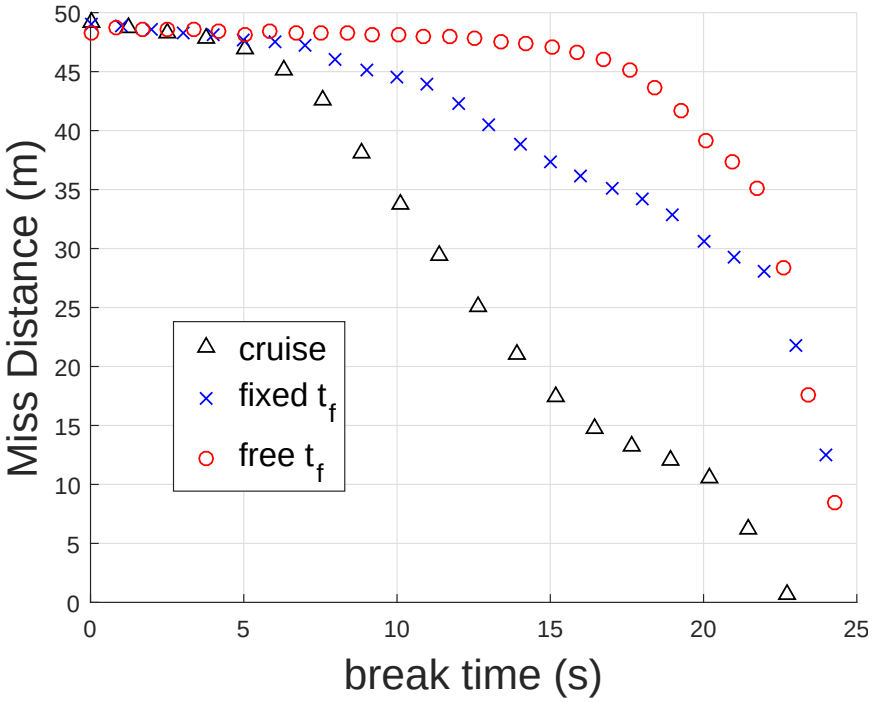

Figure 27. Results of varying the final time, t_f, for the FR problem.

terminal time in the FX problem for the same initial conditions. The CPA problem was initialized and solved at the break time for each problem, and the resulting final miss distance was recorded. The miss distance achieved by the CPA problem, hybrid FR / CPA problem, and the hybrid FX / CPA problem are compared in the polar plot in Figure 28. While neither problem achieves the best miss distances possible as represented by the CPA problem, clearly the FR problem outperforms the FX problem in general.

The difference in computational effort to calculate the full CPA problem versus the hybrid FX / CPA and FR / CPA problems is noteworthy. Computation time for an optimal control solution depends on many factors, including quality of the initial guess, number of collocation points in the initial mesh, number of segments in the initial mesh, NLP tolerance, and discretization tolerance. Therefore a quantitative comparison is difficult to make. However, qualitatively solving the CPA problem, even with a good initial guess, can require hundreds of seconds on a current desktop

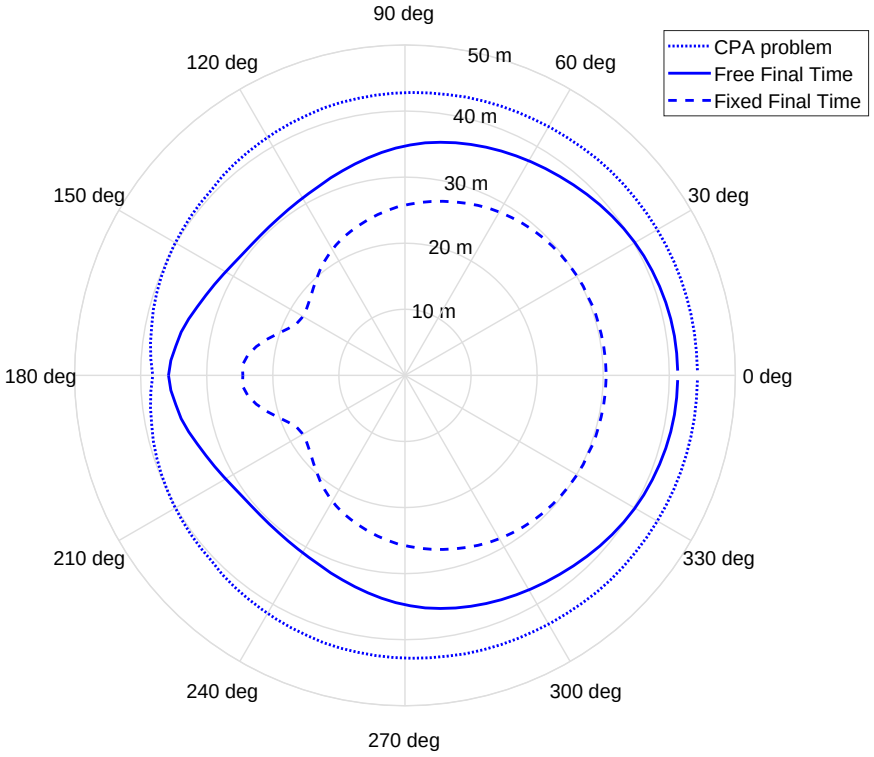

Figure 28. Comparison of final miss distance for the CPA, FR and FX problems varying the initial bearing from missile to aircraft.

computer. Solving the hybrid FX / CPA or FR / CPA problem with t_{go} of five to seven seconds requires ten to twenty seconds of computation, with the majority of this for solving the reduced CPA problem. For many cases, with a good initial guess, the FX or FR problem can be solved in under one second. In fact, much of this time is spent in the optimizer verifying the setup of the problem.

It has been mentioned previously that the maneuverability phase of the CPA problem is similar to the HGBR. If the CPA problem in the hybrid solutions is replaced by the HGBR, additional computational savings are possible, although at the expense of final miss distance. A demonstration of the performance of a hybrid FR / HGBR trajectory has been performed by cutting the FR trajectory at various times, and running the HGBR until the moment of closest approach. The miss distance was calculated and plotted versus the break time in Figure 29. It is seen that if the FR

trajectory is cut at $t_{go} = 3$ seconds, a miss distance of 30 meters is still obtained from the HGBR maneuver. This solution can be obtained in less than one second of computation time. Although the miss distance of the hybrid FR / HGBR is not as high as the hybrid FR / CPA, it is perhaps justifiable in some situations to trade performance for computational speed.

Another interesting possibility would be to replace the CPA or HGBR problem with a linear version of missile evasion. Likely, for these last few seconds of flight, the dynamics may be approximated as linear with fixed velocity. In this case, the 3 dimensional optimal evasion solution has been worked out [56], consisting of a max g pull at a roll angle in-plane with the initial collision, followed by a rapid 180 degree roll and again pulling maximum g. The moment at which to perform the roll is determined by a switching function, which is tabulated in [56] for various values of the missile's navigation gain and time constant.

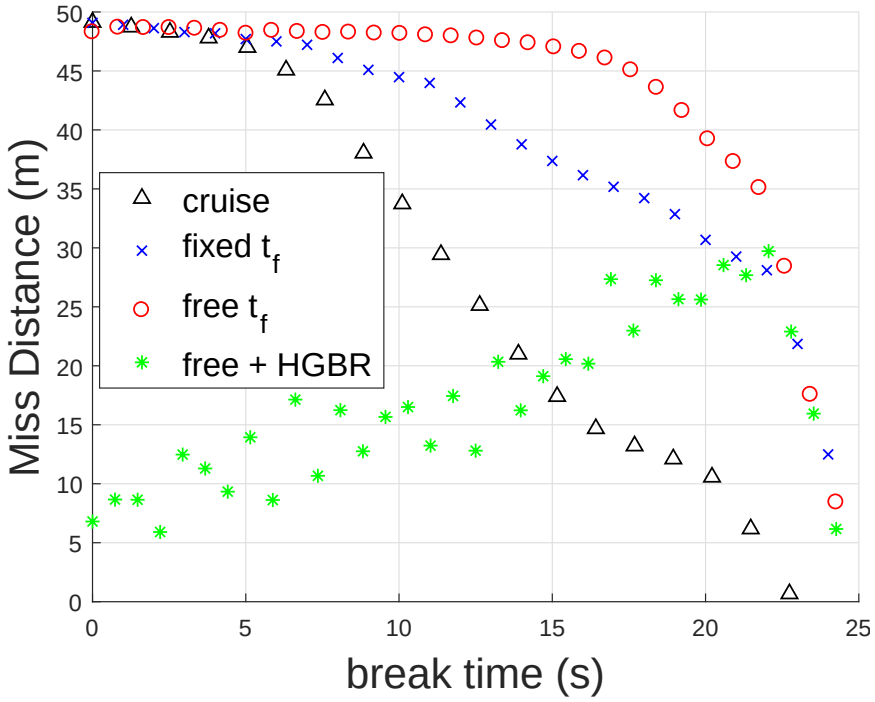

Figure 29. Results of varying the final time, t_f, for the hybrid FR / HGBR problem.

4.4 Summary

The final miss distance objective, defined by the moment of closest approach when the closing velocity reaches zero, is a logical pick when seeking an optimal evasion strategy against a single missile. However, as demonstrated here, obtaining the solution to this problem is difficult due to the high gradient which is inherent to the terminal constraint. To enable faster solutions with more reliable convergence the problem has been split in two phases, where energy and maneuverability are respectively dominant. The maneuverability phase is still represented by a CPA problem, although the solution is more easily calculated when the range has already been reduced. The energy management phase may be represented either by a fixed final time, or free final time optimal control problem. Details of the trajectories resulting from these two formulations have been provided here, primarily showing that the free final time problem seems to better manage its energy prior to the maneuverability phase. It was also demonstrated that the HGBR maneuver may be a substitute for the CPA problem, trading performance for computational speed.

For the remainder of this work it will be assumed that the FX and FR problems are surrogates for the energy management phase of the overall evasion effort. While they have been shown to be linked to the CPA objective, they also stand alone in their utility. In fact, dodging the missile using high-g maneuvers is the pilot's last resort. Implementing problems FX and FR extends the time for other types of countermeasures to be employed. Specific modifications to FR and FX could be made to accommodate these countermeasures, and optimize their employment.

In this chapter it has been assumed that the pursuer employs the PN strategy, a relatively good assumption given its popularity, the simplicity of its implementation, and its relative optimality. However, if the guidance strategy is actually unknown, it may be necessary to relax this assumption and allow that the pursuer may also

choose an optimal open-loop strategy. This leads to solving a classic pursuit-evasion problem, where the objective is zero-sum between the evader and pursuer, and the solution is a minimax. In the next chapter the semi-DCNLP method will be used to solve minimax problems mirroring the fixed and free final time one-sided problems just presented. It will also be shown that the minimax solution represents a guarantee on the cost function for an open-loop trajectory, at least for fixed final time problems.

V. Minimax Pursuit-Evasion Problems

5.1 Costate Estimation for the Minimax Initial Guess

So far, it has been assumed that the pursuer uses PN to intercept the evader. While PN is a popular choice for navigation, modern missiles are likely to employ advanced algorithms which may outperform PN. Therefore, it is interesting to assume that the pursuer is able to perfectly observe the evader, and act optimally to achieve its goal of achieving capture. In this chapter, two zero-sum games will be posed wherein both evader and pursuer act optimally, and minimax solutions will be calculated using two different methods: semi-DCNLP and Decomposition. The TPBVP representing the necessary optimality conditions contains the costate equations of both the evader and pursuer, and one set of these equations are collocated along with the states in semi-DCNLP. This means that an initial guess of the costates is required to solve the problem. Unfortunately the values taken on by the costates are non-intuitive, and the dynamics of the costates are numerically sensitive to the initial guess. This necessitates that the user generate an initial guess that is much closer to the final solution than for a normal one-sided problem. In the literature review it was remarked that this was done previously using a GA pre-conditioner. However, this method normally requires long calculation times, and is usually only an approximation. For this reason, an alternative method is presented for generating the initial guess by solving a similar one-sided problem, and using collocation based costate estimation.

Costate estimation for PS methods was reviewed in Chapter II, resulting in Equations (2.15), which relate the KKT multipliers from the NLP to the continuous costates. This mapping makes it possible to estimate costates for a minimax problem from the solution of one or more one-sided easily solved optimization problems. The specific set of one-sided problems to be solved depends on the minimax problem. A

method for generating an initial guess for the fixed final time version will be presented, but first it is necessary to define the problem to be solved.

5.2 The Fixed Final Time Minimax Problem

The zero-sum objective mirrors the one-sided objective from Equation (4.5), but with additional integral control terms for the pursuer, weighted by w_P,

$$J = r(t_f) + \int_{t_0}^{t_f} w_P \left[\left(\frac{a_{\gamma,c}}{a_M} \right)^2 + \left(\frac{a_{\chi,c}}{a_M} \right)^2 \right] - w_E \left[\left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M} \right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M} \right)^2 \right] dt, \quad (5.1)$$

where the evader's control vector is $\mathbf{u}_E = (u_\alpha, u_\mu)$ and the pursuer's control vector is $\mathbf{u}_P = (a_{\gamma,c}, a_{\chi,c})$. In the fixed final time version of the problem there is no terminal constraint other than the final time, t_f. For simplicity the path constraints on maximum g for the aircraft and missile will be relaxed, along with constraints on bank angle and angle of attack. It will be seen that for the energy phase of the problem, these are unnecessary for many solutions. Also, in order to limit the controls without actually imposing an inequality constraint, the aircraft controls will be transformed as is done in [85] via

$$\begin{aligned} u_\alpha &= \dot{\alpha}_M \sin(u_{\bar{\alpha}}\ \pi/2) \\ u_\mu &= \dot{\mu}_M \sin(u_{\bar{\mu}}\ \pi/2), \end{aligned} \quad (5.2)$$

thus u_α is only defined for $-\dot{\alpha}_M \leq u_\alpha \leq \dot{\alpha}_M$ deg/s, and u_μ is defined for $-\dot{\mu}_M \leq u_\mu \leq \dot{\mu}_M$ deg/s. In a similar fashion, the missile controls will be redefined as

$$\begin{aligned} a_{\gamma,c} &= a_M \sin(a_{\bar{\gamma}}\ \pi/2) \\ a_{\chi,c} &= a_M \sin(a_{\bar{\chi}}\ \pi/2), \end{aligned} \quad (5.3)$$

where $a_M = (35\ g)$ m/s^2, so that the missile commands are also restricted to being $-35\ g \leq a_{\gamma,c} \leq 35\ g$ m/s^2 and $-35\ g \leq a_{\chi,c} \leq 35\ g$ m/s^2. This redefinition allows

the controls to be unbounded, which will simplify the formation of the necessary conditions.

The definition of the fixed final time, two-sided minimax problem is then to choose $\mathbf{u}_E$ and $\mathbf{u}_P$ to find the minimax of the objective in Equation (5.1), subject to the dynamic constraints in Equations (3.3) and (3.11), the controls in Equations (5.2) and (5.3), and the initial conditions in Table 4. This problem will be called the Fixed Time Minimax (FXM) problem.

5.2.1 Necessary Optimality Conditions.

While the FXM problem cannot be solved directly using the PS method, it is possible to form the TPBVP using the necessary optimality conditions, and subsequently solve using the semi-DCNLP method. This will require forming two subproblems, FXM_E and FXM_P. Before describing the details of the semi-DCNLP approach, the entire TPBVP for the FXM problem will be defined. First it is important to note that the running costs shown in Equation (5.1) and the state equations given in Equations (3.3) and (3.11) are separable, as in the sense of Equation (2.41). This means that the Hamiltonian can also be separated, and therefore the adjoint equations and stationarity conditions can be too. The terminal costate constraints will link the evader and pursuer.

5.2.1.1 Costate Equations.

By applying Equation (2.44a) to Equation (3.3), the costate dynamics for the evader are found to be

$$
\begin{aligned}
\dot{\lambda}_{x_E} &= 0 \\
\dot{\lambda}_{y_E} &= 0 \\
\dot{\lambda}_{h_E} &= -\frac{\lambda_{v_E}}{m_E}\left(\frac{\partial T_E}{\partial h_E}\cos\alpha_E - \frac{\partial D_E}{\partial h_E}\right) - \frac{\lambda_{g_E}\cos\mu_E}{m_E V_E}\left(\frac{\partial T_E}{\partial h_E}\sin\alpha_E + \frac{\partial L_E}{\partial h_E}\right) \\
&\quad - \frac{\lambda_{g_E}\sin\mu_E}{m_E V_E\cos\gamma_E}\left(\frac{\partial T_E}{\partial h_E}\sin\alpha_E + \frac{\partial L_E}{\partial h_E}\right) \\
\dot{\lambda}_{v_E} &= -\lambda_{x_E}\cos\gamma_E\cos\chi_E - \lambda_{y_E}\cos\gamma_E\sin\chi_E - \lambda_{h_E}\sin\gamma_E \\
&\quad - \frac{\lambda_{V_E}}{m_E}\left(\frac{\partial T_E}{\partial V_E}\cos\alpha_E - \frac{\partial D_E}{\partial V_E}\right) + \frac{\lambda_{\gamma_E}}{m_E V_E^2}\left(\cos\mu_E(T_E\sin\alpha_E + L_E) - g m_E\cos\gamma_E\right) \\
&\quad + \frac{\lambda_{\chi_E}\sin\mu_E}{m_E V_E\cos\mu_E}\left((T_E\sin\alpha_E + L_E)/V_E - \left(\frac{\partial T_E}{\partial V_E}\cos\alpha_E + \frac{\partial L_E}{\partial V_E}\right)\right) \\
\dot{\lambda}_{\gamma_E} &= \lambda_{x_E} V_E\sin\gamma_E\cos\chi_E + \lambda_{y_E} V_E\sin\gamma_E\sin\chi_E - \lambda_{h_E} V_E\cos\gamma_E + \lambda_{V_E} g\cos\gamma_E \\
&\quad - \lambda_{\gamma_E}\frac{g\sin\gamma_E}{V_E} - \frac{\lambda_{\chi_E}\sin\mu_E\tan\gamma_E}{m_E V_E\cos\gamma_E}(T_E\sin\alpha_E + L_E) \\
\dot{\lambda}_{\chi_E} &= \lambda_{x_E} V_E\cos\gamma_E\sin\chi_E - \lambda_{y_E}\cos\gamma_E\cos\chi_E \\
\dot{\lambda}_{\alpha_E} &= \frac{\lambda_{v_E}}{m_E}\left(T_E\sin\alpha_E + \frac{\partial D_E}{\partial\alpha_E}\right) - \frac{\lambda_{\gamma_E}\cos\mu_E}{m_E V_E}\left(T_E\cos\alpha_E + \frac{\partial L_E}{\partial\alpha_E}\right) \\
&\quad - \frac{\lambda_{\chi_E}\cos\mu_E}{m_E V_E\cos\gamma_E}\left(T_E\cos\alpha_E + \frac{\partial L_E}{\partial\alpha_E}\right) \\
\dot{\lambda}_{\mu_E} &= \frac{\lambda_{\gamma_E}\sin\mu_E}{m_E V_E}(T_E\sin\alpha_E + L_E) - \frac{\lambda_{\chi_E}\cos\mu_E}{m_E V_E\cos\gamma_E}(T_E\cos\alpha_E + L_E).
\end{aligned}
\tag{5.4}
$$

Note the above costate equations contain partial derivatives of thrust, drag, and lift with respect to altitude, velocity, and angle of attack. For this reason much attention was given to ensure the approximate fits to the tabulated lift, drag, and thrust data had continuous derivatives. The lift and drag expressions are functions of Mach

number, thus the partials with respect to altitude and velocity must be calculated using the chain rule. The need to calculate these partials is one drawback of the semi-DCNLP method, as it complicates finding solutions to problems with tabulated data. The x_E and y_E costates are constant, meaning that they do not need to be modeled as differential equations. This helps reduce the size of the overall TPBVP.

Application of Equation (2.44b) to the state Equations (3.11) yields the pursuer costate dynamics,

$$\begin{aligned}
\dot{\lambda}_{x_P} &= 0 \\
\dot{\lambda}_{h_P} &= 0 \\
\dot{\lambda}_{h_P} &= \frac{\lambda_{V_P}}{m_P}\frac{\partial D_P}{\partial h_P} \\
\dot{\lambda}_{V_P} &= -\lambda_{x_P}\cos\gamma_P\cos\chi_P - \lambda_{y_P}\cos\gamma_P\sin\chi_P - \lambda_{h_P}\sin\gamma_P + \frac{\lambda_{V_P}}{m_P}\frac{\partial D_P}{\partial V_P} \\
&\quad + \frac{\lambda_{\gamma_P}(a_{\gamma_P} - g\cos\gamma_P)}{V_P^2} + \frac{\lambda_{\chi P}a_{\chi P}}{V_P^2\cos\gamma_P} \\
\dot{\lambda}_{\gamma_P} &= \lambda_{x_P}V_P\sin\gamma_P\cos\chi_P + \lambda_{y_P}V_P\sin\gamma_P\sin\chi_P - \lambda_{h_P}V_P\cos\gamma_P + \lambda_{V_P}g\cos\gamma_P \\
&\quad - \frac{\lambda_{\gamma_P}g\sin\gamma_P}{V_P} - \frac{\lambda_{\chi P}a_\gamma\sin\gamma_P}{V_P\cos{\gamma_P}^2} \\
\dot{\lambda}_{\chi_P} &= \lambda_{x_P}V_P\cos\gamma_P\sin\chi_P - \lambda_{y_P}V_P\cos\gamma_P\cos\chi_P \\
\dot{\lambda}_{a_\gamma} &= \frac{\lambda_{V_P}}{m_P}\frac{\partial D_P}{\partial a_\gamma} - \frac{\lambda_{\gamma_P}}{V_P} + \frac{\lambda_{a_\gamma}}{\tau} \\
\dot{\lambda}_{a_\chi} &= \frac{\lambda_{V_P}}{m_P}\frac{\partial D_P}{\partial a_\chi} - \frac{\lambda_{\chi P}}{V_P\cos\gamma_P} + \frac{\lambda_{a_\chi}}{\tau}.
\end{aligned} \tag{5.5}$$

Again, the x_P and y_P costates are constant, meaning that the total number of costate equations is only 12, rather than 16, with four free variables to be determined. The partial derivatives of drag with respect to altitude, velocity, and accelerations are relatively easy to calculate given Equation (3.13).

5.2.1.2 Stationarity Conditions.

The stationarity conditions require that the controls minimize the Hamiltonian. When the controls are unbounded, as they are assumed to be here, the condition simplifies to being that the partial derivative of the Hamiltonian with respect to the control must equal zero. For the current problem, the stationarity conditions become

$$\begin{aligned}
\lambda_{\alpha_E}\dot{\alpha}_M\frac{\pi}{2}\cos(u_{\bar{\alpha}}\ \pi/2) + w_E\pi\cos(u_{\bar{\alpha}}\ \pi/2)\sin(u_{\bar{\alpha}}\ \pi/2) = 0 \\
\lambda_{\mu_E}\dot{\mu}_M\frac{\pi}{2}\cos(u_{\bar{\mu}}\ \pi/2) + w_E\pi\cos(u_{\bar{\mu}}\ \pi/2)\sin(u_{\bar{\mu}}\ \pi/2) = 0,
\end{aligned} \tag{5.6}$$

which can be solved for the control variables, $u_{\bar{\alpha}}$ and $u_{\bar{\mu}}$, to be

$$u_{\bar{\alpha}} = \begin{cases} 1 & \lambda_{\alpha_E} \leq -\dfrac{2w_E}{\dot{\alpha}_M} \\ \dfrac{2}{\pi}\arcsin\left(\dfrac{\lambda_{\alpha_E}\dot{\alpha}_M}{2w_E}\right) & -\dfrac{2w_E}{\dot{\alpha}_M} < \lambda_{\alpha_E} < \dfrac{2w_E}{\dot{\alpha}_M} \\ -1 & \lambda_{\alpha_E} \geq \dfrac{2w_E}{\dot{\alpha}_M} \end{cases}, \tag{5.7}$$

and

$$u_{\bar{\mu}} = \begin{cases} 1 & \lambda_{\mu_E} \leq -\dfrac{2w_E}{\dot{\mu}_M} \\ \dfrac{2}{\pi}\arcsin\left(\dfrac{\lambda_{\mu_E}\dot{\mu}_M}{2w_E}\right) & -\dfrac{2w_E}{\dot{\mu}_M} < \lambda_{\mu_E} < \dfrac{2w_E}{\dot{\mu}_M} \\ -1 & \lambda_{\mu_E} \geq \dfrac{2w_E}{\dot{\mu}_M} \end{cases}. \tag{5.8}$$

Control stationarity applied to the pursuer's unbounded controls gives the expressions

$$\begin{aligned}
\lambda_{\gamma_P}a_M\frac{\pi}{2\tau}\cos(u_{\bar{\gamma}}\ \pi/2) + w_P\pi\cos(u_{\bar{\gamma}}\ \pi/2)\sin(u_{\bar{\gamma}}\ \pi/2) = 0 \\
\lambda_{\chi_P}a_M\frac{\pi}{2\tau}\cos(u_{\bar{\chi}}\ \pi/2) + w_P\pi\cos(u_{\bar{\chi}}\ \pi/2)\sin(u_{\bar{\chi}}\ \pi/2) = 0,
\end{aligned} \tag{5.9}$$

which can also be solved for the control variables, $u_{\bar{\gamma}}$ and $u_{\bar{\chi}}$, to give

$$u_{\bar{\gamma}} = \begin{cases} 1 & \lambda_{a_\gamma} \leq -\dfrac{2\tau w_P}{a_M} \\ \dfrac{2}{\pi}\arcsin\left(\dfrac{\lambda_{a_\gamma} a_M}{2\tau w_P}\right) & -\dfrac{2\tau w_P}{a_M} < \lambda_{a_\gamma} < \dfrac{2\tau w_P}{a_M} \\ -1 & \lambda_{a_\gamma} \geq \dfrac{2\tau w_P}{a_M} \end{cases}, \tag{5.10}$$

and

$$u_{\bar{\chi}} = \begin{cases} 1 & \lambda_{a_\chi} \leq -\dfrac{2\tau w_P}{a_M} \\ \dfrac{2}{\pi}\arcsin\left(\dfrac{\lambda_{a_\chi} a_M}{2\tau w_P}\right) & -\dfrac{2\tau w_P}{a_M} < \lambda_{a_\chi} < \dfrac{2\tau w_P}{a_M} \\ -1 & \lambda_{a_\chi} \geq \dfrac{2\tau w_P}{a_M} \end{cases}. \tag{5.11}$$

The usefulness of the squared control terms within the running cost of the objective in Equation (5.1) can now be seen. Without these terms, the controls would not appear in Equations (5.6) and (5.9), instead there would be a singular arc where the control is undefined. Including the control terms in the running cost has the effect of allowing the controls to be expressed in terms of the costates in the TPBVP, which can then be substituted directly into the equations of motion to form a reduced set of differential equations. Additionally, it has been observed by the author that numerical chatter in the collocation solution is reduced by including squared control terms in the running cost.

5.2.1.3 Terminal Costate Conditions.

The scalar objective in Equation (5.1) is the magnitude of the relative position vector given as

$$r(t_f) = \sqrt{(x_E(t_f) - x_P(t_f))^2 + (y_E(t_f) - y_P(t_f))^2 + (h_E(t_f) - h_P(t_f))^2}. \tag{5.12}$$

Applying the terminal costate necessary condition from Equation (2.47) gives

$$\begin{aligned}
\lambda_{x_E}(t_f) &= -\frac{x_E(t_f) - x_P(t_f)}{r(t_f)} \\
\lambda_{y_E}(t_f) &= -\frac{y_E(t_f) - y_P(t_f)}{r(t_f)} \\
\lambda_{h_E}(t_f) &= -\frac{h_E(t_f) - h_P(t_f)}{r(t_f)} \\
\lambda_{v_E}(t_f) &= \lambda_{\gamma_E}(t_f) = \lambda_{\chi_E}(t_f) = \lambda_{\alpha_E}(t_f) = \lambda_{\mu_E}(t_f) = 0,
\end{aligned} \tag{5.13}$$

for the evader, and

$$\begin{aligned}
\lambda_{x_P}(t_f) &= -\frac{x_E(t_f) - x_P(t_f)}{r(t_f)} \\
\lambda_{y_P}(t_f) &= -\frac{y_E(t_f) - y_P(t_f)}{r(t_f)} \\
\lambda_{h_P}(t_f) &= -\frac{h_E(t_f) - h_P(t_f)}{r(t_f)} \\
\lambda_{v_P}(t_f) &= \lambda_{\gamma_P}(t_f) = \lambda_{\chi_P}(t_f) = \lambda_{a_\gamma}(t_f) = \lambda_{a_\chi}(t_f) = 0,
\end{aligned} \tag{5.14}$$

for the pursuer. Since λ_{x_E}, λ_{y_E}, λ_{x_P}, and λ_{y_P} are constants, they are completely defined by Equations (5.13) and (5.14).

5.2.2 Initial Guess Using One-Sided Costate Estimation.

One weakness of the semi-DCNLP method is the need to generate an initial guess for the nonintuitive costates. While several authors used a GA pre-conditioner to produce a guess for the costates, it is possible to simply estimate them by solving a one-sided optimal control problem. The FX problem solved in Chapter IV is similar to the current minimax problem, except for the assumption of PN for the pursuer. One result of solving the FX problem is a set of Lagrange multipliers from the NLP solver. The continuous costates can then be estimated from these multipliers using Equations (2.15).

The sign (+/-) of the terminal value of each costates is linked to the scalar objective through Equation (2.47). Thus the sign of the terminal value of the costates for the FX problem are set by the evader's objective, which is to maximize the final distance between the evader and pursuer. Using the solution of problem FX to estimate the costates works well for the evader's costates in the FXM problem, because they share a common terminal objective. However, the objective of the pursuer in the FXM problem is instead to *minimize* this distance, thus the terminal value of the costates from the FX problem cannot be expected to match those for the FXM problem. For this reason it is helpful to define another one-sided problem, FX_P, which has the objective to minimize the final distance between the evader and pursuer.

For the FX problem, PN was assigned as a suboptimal behavior for the pursuer. For FX_P, a simple suboptimal behavior for the evader is to fly to the evader's final position from the FX problem. Under this assumption, it is not necessary to model the evader's dynamics, as this was already done in problem FX. Problem FX_P is solved using only the pursuer's dynamics, with the objective of minimizing the pursuer's final distance away from the evader's final position from problem FX. The solution to problem FX_P then produces a good initial guess at the states, controls, and estimated costates for the pursuer in the FXM problem. Thus, the states, costates, and controls for the evader are estimated using the solution to problem FX, while the states, costates, and controls for the pursuer are estimated from problem FX_P. Estimates of the evader's costates generated using the FX problem are shown in Figure 30 compared to the true values which will be calculated via semi-DCNLP. The estimated evader costates from the FX problem are reasonably close to the final values. The pursuer costates from the FX problem are not close to the correct values, as shown in Figure 31. The pursuer costates generated using the FX_P problem, however, are very close to the final correct value.

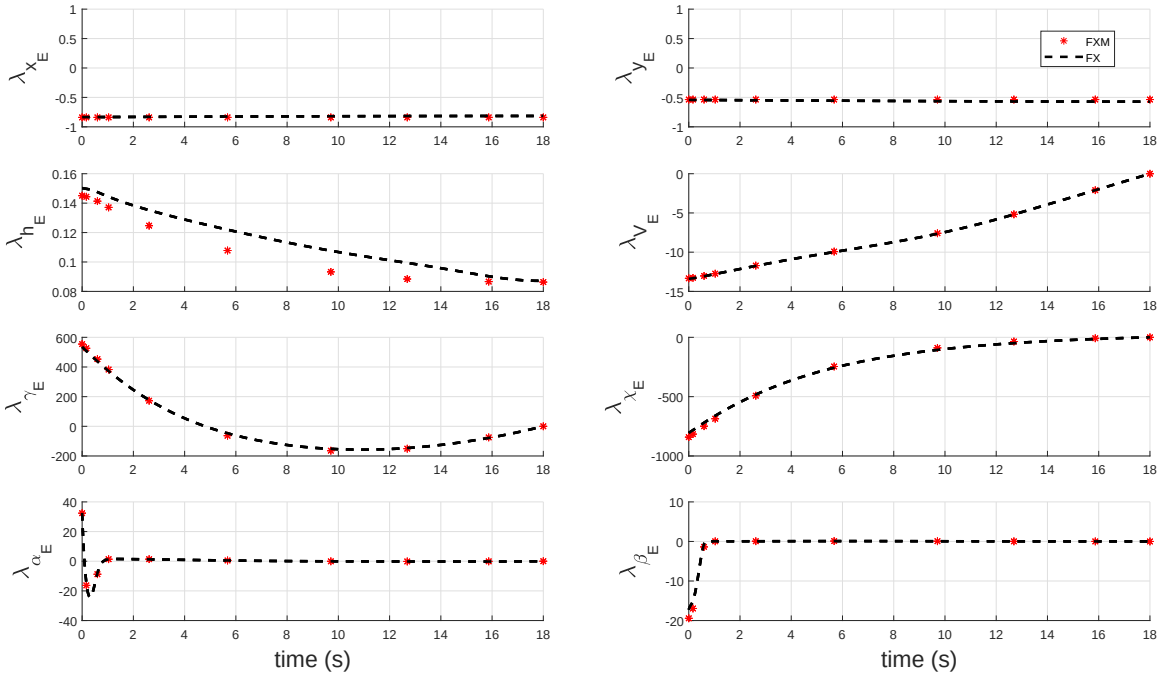

Figure 30. Evader costates estimated from problem FX, compared with the solution to problem FXM.

5.2.3 Fixed Final Time Minimax Problem - semi-DCNLP.

The state Equations (3.3) and (3.11), the costate Equations (5.4) and (5.5), the terminal constraints (5.13) and (5.14), and the initial conditions in Table 4 define the TPBVP representing the necessary optimality conditions of the FXM problem. Using the initial guess obtained by solving problems FX and FX_P, it is possible to solve FXM using semi-DCNLP in two different ways; one by collocating the evader's controls, FXM_E, and the other by collocating the pursuer's controls, FXM_P.

To pose problem FXM_E, the costate equations corresponding to the pursuer are included in the collocation as states, and their terminal values are constrained by Equation (5.14). Rather than collocating the pursuer's controls, the stationarity conditions are used to eliminate them from the problem via Equations (5.10) and (5.11). With the necessary conditions built into the problem, the integral term in the objective function corresponding to the pursuer's controls has been accounted

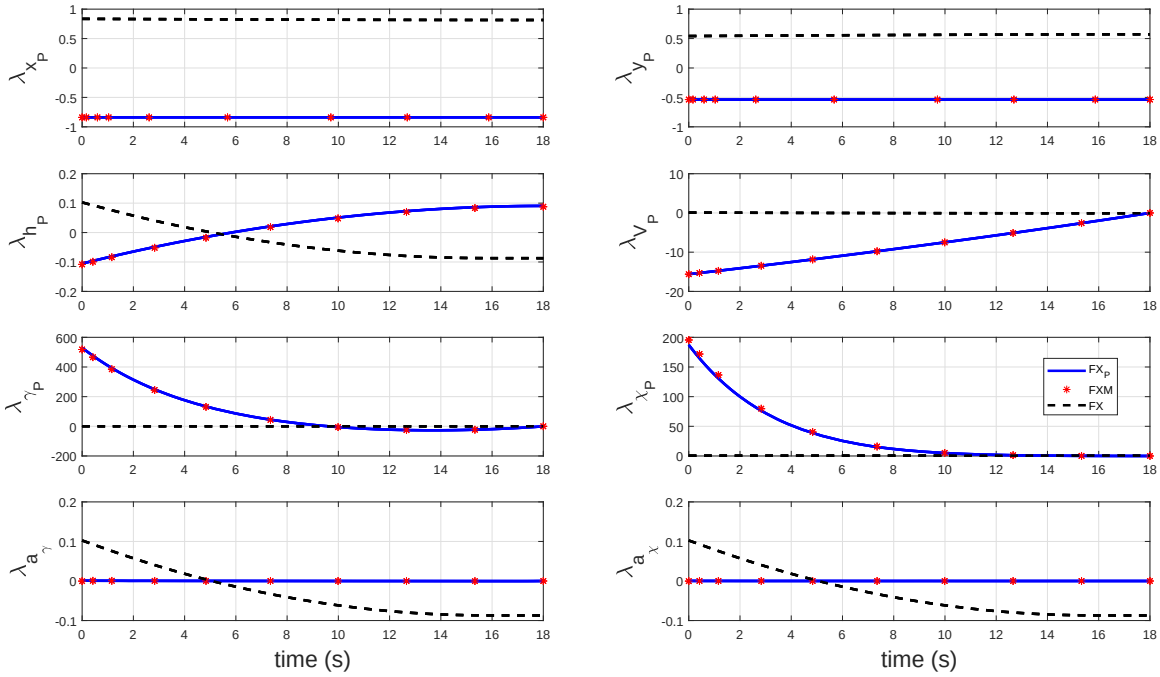

Figure 31. Pursuer costates estimated from problem FX and FX$_P$, compared with the solution to problem FXM.

for, and may be dropped. Finally, the minimization part of the minimax objective is accomplished by the inclusion of the preceding necessary conditions, thus the modified objective needs only to be maximized. The objective to maximize for problem FXM$_E$ is

$$J = r(t_f) - w_E \int_{t_0}^{t_0} \left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M}\right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M}\right)^2 dt. \tag{5.15}$$

Since GPOPS-II expects to minimize the objective, Equation (5.15) is coded into the software with the opposite sign.

To pose problem FXM$_P$, the evader's costates are included as states in the collocation, along with their terminal constraints. The evader's controls are eliminated by substitution of the stationarity conditions, Equations (5.7) and (5.8), into the evader's dynamics. The running cost associated with the evader's controls are eliminated from the objective, along with the maximization. Thus, the modified objective for problem

FXM_P coded into GPOPS-II is

$$J = r(t_f) + w_P \int_{t_0}^{t_f} \left(\frac{a_{\gamma,c}}{a_M}\right)^2 + \left(\frac{a_{\chi,c}}{a_M}\right)^2 dt. \tag{5.16}$$

It is now possible to formally define the two sides of the semi-DCNLP method of solving the pursuit-evasion game FXM.

The definition of FXM_E is to choose the controls $u_{\bar{\alpha}}$ and $u_{\bar{\mu}}$ to maximize the objective in Equation (5.15), subject to the dynamics in Equations (3.3) and (3.11), the evader's control definition in Equation (5.2), the pursuer's costate dynamics in Equation (5.5), the terminal costate conditions for the pursuer in Equation (5.14), and the pursuer's stationarity conditions in Equations (5.10) and (5.11).

The definition of FXM_P is to choose the controls $u_{\bar{\gamma}}$ and $u_{\bar{\chi}}$ to maximize the objective in Equation (5.16), subject to the dynamics in Equations (3.3) and (3.11), the pursuer's control definition in Equation (5.3), the evader's costate dynamics in Equation (5.4), the terminal costate conditions for the evader in Equation (5.13), and the evader's stationarity conditions in Equations (5.7) and (5.8).

While technically all the necessary conditions for problem FXM will be satisfied by solving either FXM_E or FXM_P, it is helpful to solve both problems in order to ensure that the true minimax solution has been obtained. It is possible that either solution may be a local minimum, but this issue may be detected by solving both problems and comparing the value of J in Equation (5.1). Additionally, it is helpful to compare the state trajectories of each in order to visualize small differences in the event of numerical chatter in the control. Figure 32 highlights the entire process of solving problem FXM via semi-DCNLP, from solution of the one-sided guesses FX and FX_P, to comparison of the final objective values of solutions to FXM_E and FXM_P.

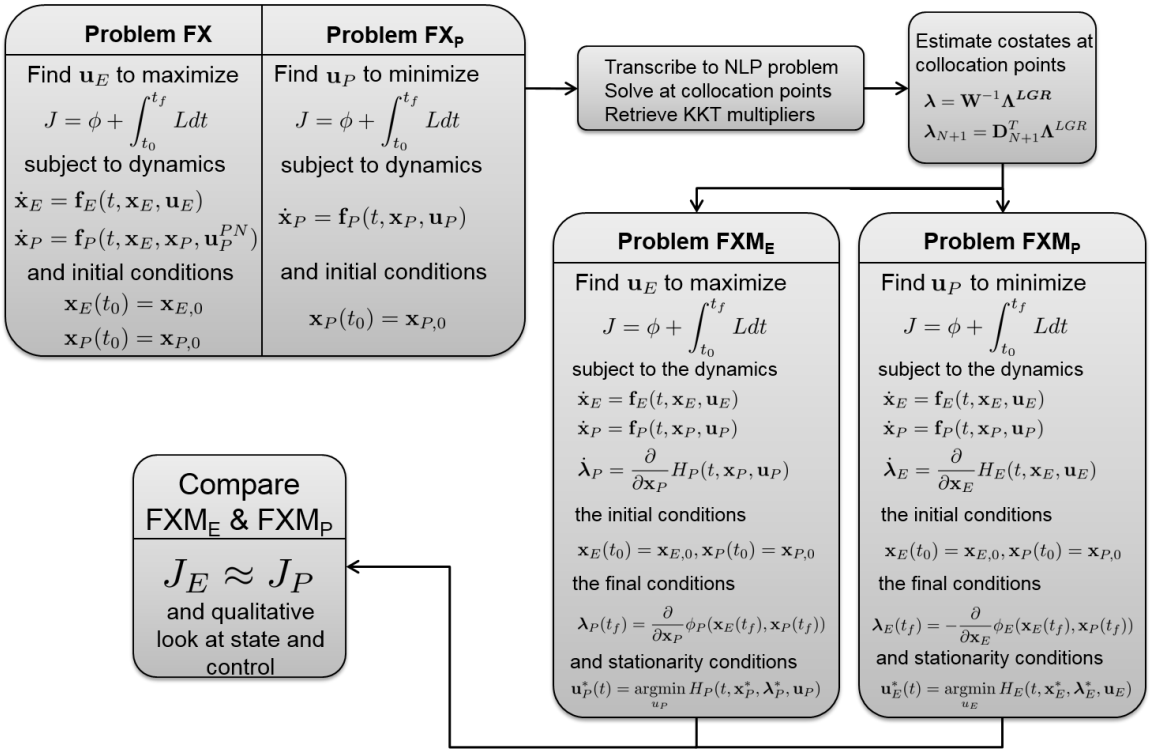

Figure 32. Solution process to problem FXM via semi-DCNLP.

This process was followed to solve problem FXM. The objective functions actually coded into GPOPS-II for problems FXM_E and FMX_P are different from the FXM objective in Equation (5.1), in that each is missing the running cost of the other. However, these integrals can be calculated post-process via the collocation rules of the PS method. This is most easily accomplished by defining the extra integrand in the software and allowing it to be computed along with the solution, and simply displaying it with the output. When this is done the value of the FXM objective calculated by solving FXM_E is 6,043.9282 meters, while the objective found by solving FXM_P is 6,043.9776 meters. The error between the two, defined by the difference divided by the evader's objective, is only 8.17×10^{-6}, very nearly the NLP tolerance of each solution (1×10^{-6}). Because the same solution has been achieved by both sides of the problem, it can be considered that their trajectories represent the minimax.

Because the results from FXM_E and FXM_P are so close, it is impossible to visually distinguish the two trajectories. Therefore the altitude and bank angle of FXM_E are shown compared with the initial guess FX in Figure 33. Most notably, the value of the objective from problem FX is 6,063.3265 meters, while the objective value of FX_P is 6,042.3198 meters. The minimax value is, not surprisingly, between these two. This is because in problem FX, the pursuer uses PN, which is not the optimal strategy, thus the evader improves its objective. In problem FX_P, the pursuer's optimal control is found. However, although in FX_P the evader used the optimal trajectory from problem FX, it is no longer an optimal strategy against the optimal pursuer. Thus the minimax trajectory represents the value of the objective when both the evader and the pursuer behave optimally, and if either player does not choose the minimax strategy it results in a detriment to that player, and a benefit to their adversary.

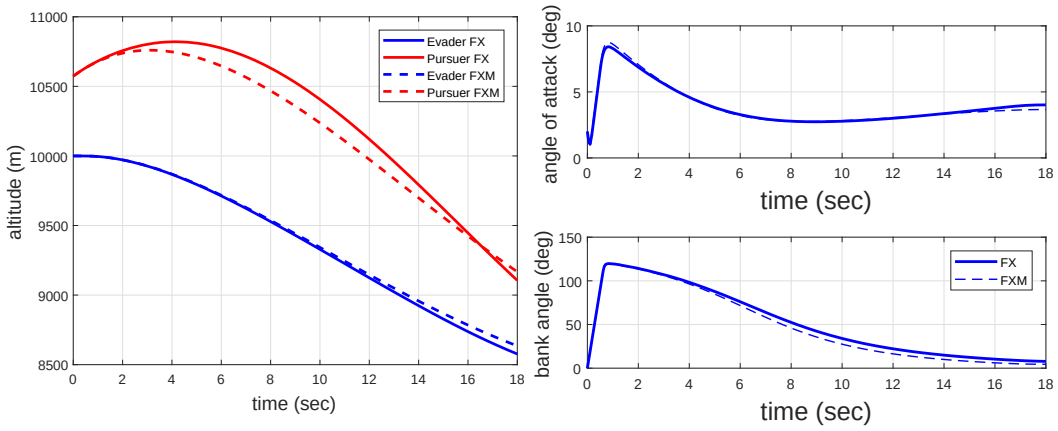

Figure 33. Altitude (left) and angle of attack and bank angle (right) of the Fixed Time Minimax trajectory compared to the one-sided Fixed Time trajectory.

5.2.4 Implementing a State Constraint.

One feature that does not appear in the original work by Horie and Conway [85] is how to handle a pure state inequality constraint in problem FXM. The controls were bounded by transforming them via the sine function which has range between -1 and

+1. This transformation cannot be done for pure state inequality constraints, which appear in many realistic optimal control problems. For a direct method, pure state constraints are applied simply by imposing a constraint within the NLP at the collocation points. An appropriately dense mesh will then ensure the inequality constraint is met. However, the semi-DCNLP method mixes the direct and indirect methods. In indirect methods, a pure state inequality constraint is handled by first determining when, if at all, the state reaches the boundary. In order to ensure the state does not further penetrate the boundary, the derivative of the state is constrained, often creating a new mixed inequality constraint. This then imposes a jump condition on the costate. Afterward, it is also necessary to determine if and when the state leaves the boundary, and remove the mixed inequality constraint [115, 116]. This complicates the semi-DCNLP method because if a state constraint is placed on the pursuer, problem FXM_E must implement these jump conditions in order to accurately write the costate equations. Inclusion of the jump conditions within the PS method is very complicated, especially when multiple state constraints must be checked, resulting in many separate jump conditions, which may or may not actually be necessary in the problem.

Soft state constraints, namely, penalty and barrier methods, present a way to avoid having to implement jump conditions on the costates within semi-DCNLP. In these methods the problem definition is changed by removing the hard constraint and instead appending a penalty or barrier to the objective function [21]. These penalties or barriers are typically functions that equal zero whenever the inequality constraint is met, but increase rapidly when the constraint is active. Although the problem is modified, the value of the objective function may only vary slightly from the original problem, provided the penalty is properly tuned. Because the NLP solvers used in GPOPS-II rely on gradient calculations, it is helpful if the penalty function has a

continuous derivative. One candidate is the softplus function, which is typically used as an activation function in neural networks [117]. If, for example, it is desired to penalize altitudes below zero, the softplus penalty would take the form

$$g(h_E) = \ln(1 + e^{-h_E}), \tag{5.17}$$

which is small for $h_E \geq 0$, increases rapidly for $h_E < 0$, and has smoothly continuous derivatives for all h_E.

When a softplus function representing a state inequality constraint is added to the objective function in Equation (5.1), an additional term appears in the altitude costate dynamics in Equation (5.4), becoming

$$\begin{aligned}\dot{\lambda}_{h_E} = \frac{e^{-h_E}}{e^{-h_E}+1} - \frac{\lambda_{v_E}}{m_E}\left(\frac{\partial T_E}{\partial h_E}\cos\alpha_E - \frac{\partial D_E}{\partial h_E}\right) - \frac{\lambda_{g_E}\cos\mu_E}{m_E V_E}\left(\frac{\partial T_E}{\partial h_E}\sin\alpha_E + \frac{\partial L_E}{\partial h_E}\right) \\ - \frac{\lambda_{g_E}\sin\mu_E}{m_E V_E\cos\gamma_E}\left(\frac{\partial T_E}{\partial h_E}\sin\alpha_E + \frac{\partial L_E}{\partial h_E}\right).\end{aligned} \tag{5.18}$$

The costate boundary conditions in Equation (5.14) and the stationarity conditions in Equation (5.10) are unaffected. Because only the costate dynamics are affected, the overall size of the NLP problem does not change, whereas adding an inequality constraint within a collocation method will increase the size of the NLP problem by the number of nodes.

A modified version of FXM, where the initial altitude of the evader is only 1,000 meters, has been solved using the semi-DCNLP method using the softplus penalty to keep the evader from hitting the ground. The resulting trajectory is shown in Figure 34. It is interesting that although the unconstrained trajectory would have descended several hundred meters below $h_E = 0$, the penalty does not cause the evader to fly exactly at ground level, but instead results in the evader arriving at

exactly ground level at the final time. This matches the behavior from problem FX when the trajectory has a hard constraint.

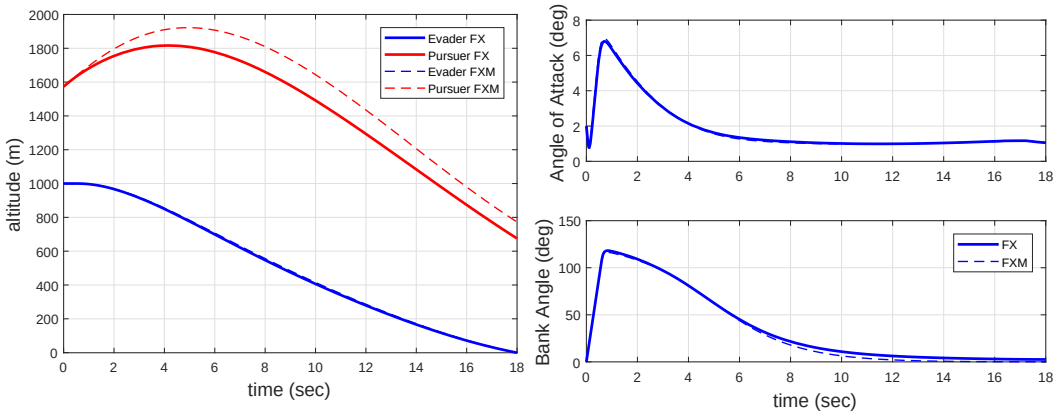

Figure 34. Altitude (left), angle of attack and bank angle (right) of the FXM trajectory compared to the one-sided FX trajectory when altitude is constrained to be greater than zero using the softplus penalty function.

5.2.5 Fixed Final Time Minimax Problem - Indirect Transcription.

It is possible to solve problem FXM in another way, where instead of collocating the costates of one player and allowing the objective and control of the other player to be solved directly, both sets of costates and corresponding terminal conditions are collocated. This eliminates the controls and the objective from the problem entirely, because they are represented by the necessary optimality conditions being enforced by the NLP. In fact, the Indirect Transcription method, as discussed in Chapter II, simply solves the TPBVP via collocation.

To implement Indirect Transcription, the state and both sets of costate equations are coded as states in GPOPS-II, while the terminal costate equations are coded as endpoint constraints. No controls are defined, and the objective is set to zero. The initial guess is formed by solving problem FX and FX_P, estimating the costates of the evader and pursuer from the respective solutions, and joining them together into a single trajectory. Because the x and y costates are constant for both evader and

pursuer, and their values are defined by Equations (5.13) and (5.14), they do not need to be included as states. Thus, 28 states are collocated: 16 actual states and 12 costates. Only two endpoint constraints must be defined in GPOPS-II, one for each of the h costates.

Once the solution has been obtained, it is possible to calculate the controls post process using the stationarity conditions in Equations (5.7), (5.8), (5.10), and (5.11). Predictably, because it collocates all the sensitive costate equations, the solution to FXM using Indirect Transcription requires a much finer grid than either the FXM_E or FXM_P problems, and typically longer computation time. The objective value obtained using Indirect Transcription was 6,044.2221 meters, which is an error of 4.87×10^{-5} compared to FXM_E, slightly larger than the error between FXM_E and FXM_P already reported. The trajectory is qualitatively very similar to FXM_E and FXM_P, and thus will not be reproduced.

5.3 The Free Final Time Minimax Problem

In the previous chapter, it was demonstrated that the one-sided free final time problem (FR) achieved superior results to the fixed final time problem (FX) when used during the energy phase of a medium range missile evasion scenario. In this section, the Free Final Time Minimax problem (FRM) will be posed and solved.

Like for the fixed final time problem, the objective for the FRM problem contains integral terms which serve to conserve control energy, dampen numerical chatter, and form the TPBVP. The objective is

$$J = t_f + w_P \int_{t_0}^{t_f} \left(\frac{a_{\gamma,C}}{a_M g}\right)^2 + \left(\frac{a_{\chi,C}}{a_M g}\right)^2 dt - w_E \int_{t_0}^{t_f} \left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M}\right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M}\right)^2 dt. \quad (5.19)$$

5.3.1 Necessary Optimality Conditions.

The necessary optimality conditions for the FRM problem share some similarities with the FXM problem. The state Equations (3.3) and (3.11), the costate Equations (5.4) and (5.5), and stationarity conditions given by Equations (5.7), (5.8), (5.10) and (5.11) are identical. However, the terminal costate conditions are very different.

5.3.1.1 Terminal Costate Conditions.

The terminal conditions for a general free final time problem are given by Equations (2.48a) - (2.48d). The terminal surface is modified from $r(t_f) = 0$ to the equivalent condition that

$$\begin{aligned} x_E - x_P &= 0 \\ y_E - y_P &= 0 \\ h_E - h_P &= 0. \end{aligned} \tag{5.20}$$

Of the evader states, only x_E, y_E, and h_E appear in the expressions for the terminal surface, thus for the evader

$$\begin{aligned} \lambda_{x_E}(t_f) &= -\nu_x \\ \lambda_{y_E}(t_f) &= -\nu_y \\ \lambda_{h_E}(t_f) &= -\nu_h, \end{aligned} \tag{5.21}$$

where ν_x, ν_y, and ν_h are multipliers with unknown values. For the rest of the costates, the condition becomes

$$\lambda_{v_E}(t_f) = \lambda_{\gamma_E}(t_f) = \lambda_{\chi_E}(t_f) = \lambda_{\alpha_E}(t_f) = \lambda_{\mu_E}(t_f) = 0. \tag{5.22}$$

For the pursuer, the x_P, y_P, and h_P terminal costates are

$$\begin{aligned} \lambda_{x_P}(t_f) &= \nu_x \\ \lambda_{y_P}(t_f) &= \nu_y \\ \lambda_{h_P}(t_f) &= \nu_h, \end{aligned} \tag{5.23}$$

and again the rest of the terminal costates are

$$\lambda_{v_P}(t_f) = \lambda_{\gamma_P}(t_f) = \lambda_{\chi_P}(t_f) = \lambda_{a_\gamma}(t_f) = \lambda_{a_\chi}(t_f) = 0. \tag{5.24}$$

5.3.1.2 Transversality.

A free final time problem requires a transversality condition, a constraint to match the unknown final time. Equation (2.48d) applied to the current problem gives

$$H_E(t_f) + H_P(t_f) + 1 = 0, \tag{5.25}$$

where the parts of the Hamiltonian at the final time for the evader and pursuer are

$$\begin{aligned} H_E(t_f) &= \lambda_{x_E}(t_f)\dot{x}_E(t_f) + \lambda_{y_E}(t_f)\dot{y}_E(t_f) + \lambda_{h_E}(t_f)\dot{h}_E(t_f) \\ H_P(t_f) &= \lambda_{x_P}(t_f)\dot{x}_P(t_f) + \lambda_{y_P}(t_f)\dot{y}_P(t_f) + \lambda_{h_P}(t_f)\dot{h}_P(t_f). \end{aligned} \tag{5.26}$$

The idea of matching the number of constraints with the number of unknowns is seen in [4], where each constraint equation is balanced with a variable or constant of integration. In this case, there are 16 known initial states, 16 equations for the terminal costates, 3 terminal constraint equations and 1 transversality equation. There are 16 unknown terminal states, 16 unknown initial costates, 3 unknown multipliers, ν_x, ν_y, and ν_h, and 1 unknown terminal time. Thus the 36 constraints match the 36 unknowns, and the TPBVP is tractable.

5.3.2 Free Final Time Minimax Problem - semi-DCNLP.

The state Equations (3.3) and (3.11), the costate Equations (5.4) and (5.5), the terminal constraint Equations (5.21 - 5.24), the terminal surface Equations (5.20), the transversality Equation (5.25), and the initial conditions in Table 4 define the TPBVP representing the necessary optimality conditions of the FRM problem.

In the literature, the semi-DCNLP method was applied to a free final time problem of two aircraft in a dogfight [85, 90]. Only one side of the problem was solved, where the pursuer's necessary conditions were collocated in the problem. However, there is an issue with this technique that did not appear in the FXM problem. In the FXM formulation, the terminal costate conditions in Equation (5.13) and (5.14) do not contain the unknown variables, ν_x, ν_y, and ν_h. Instead they are completely defined by the terminal states.

However, in the FRM problem, the terminal costate expressions do contain ν_x ν_y, and ν_h. It is possible to eliminate them from the Equations (5.21)and (5.23) by relating the evader and pursuer terminal costates, as in

$$\begin{aligned} \lambda_{x_E}(t_f) + \lambda_{x_P}(t_f) &= 0 \\ \lambda_{y_E}(t_f) + \lambda_{y_P}(t_f) &= 0 \\ \lambda_{h_E}(t_f) + \lambda_{h_P}(t_f) &= 0. \end{aligned} \tag{5.27}$$

Unfortunately, both evader and pursuer costates must be present in order to enforce these conditions. Further, the transversality condition also includes both evader and pursuer costates, although application of Equation (5.27) can reduce this dependency to either only the evader or pursuer. In the semi-DCNLP method, only one set of the costates is directly represented in each problem. Because of this, in references [90], [85], [87], and [88], the terminal costate and transversality conditions are intentionally neglected. As will be seen, this oversight can cause problems for the current

problem. While it is possible to compute a solution without the terminal costate and transversality conditions, there is no guarantee that the solution will even be a candidate trajectory, because the necessary optimality conditions have not been met.

In order to illustrate this problem, incomplete definitions of FRM_E and FRM_P as provided in the references will be used. First, for FRM_E, the objective to maximize is

$$J = t_f - w_E \int_{t_0}^{t_f} \left(\frac{\dot{\alpha}_E}{\dot{\alpha}_M}\right)^2 + \left(\frac{\dot{\mu}_E}{\dot{\mu}_M}\right)^2 dt, \tag{5.28}$$

while the objective to minimize for FRM_P is

$$J = t_f + w_P \int_{t_0}^{t_f} \left(\frac{a_{\gamma,C}}{a_M g}\right)^2 + \left(\frac{a_{\chi,C}}{a_M g}\right)^2 dt. \tag{5.29}$$

The definition of FRM_E is to choose the controls $u_{\bar{\alpha}}$ and $u_{\bar{\mu}}$ to maximize the objective in Equation (5.28), subject to the dynamics in Equations (3.3) and (3.11), the evader's control definition in Equation (5.2), the pursuer's costate dynamics in Equation (5.5), the incomplete set of terminal costate conditions for the pursuer in Equation (5.24), and the pursuer's stationarity conditions in Equations (5.10) and (5.11). Note that due to the lack of evader costates, it is impossible to enforce the costate conditions in Equation (5.27).

The definition of FRM_P is to choose the controls $u_{\bar{\gamma}}$ and $u_{\bar{\chi}}$ to maximize the objective in Equation (5.29), subject to the dynamics in Equations (3.3) and (3.11), the pursuer's control definition in Equation (5.3), the evader's costate dynamics in Equation (5.4), the incomplete terminal costate conditions for the evader in Equation (5.22), and the evader's stationarity conditions in Equations (5.7) and (5.8). Again, it is not possible to enforce the terminal costate conditions in Equation (5.27).

As described for the case of fixed final time, it is possible to generate an initial guess using a similar one-sided optimal control problem by estimating the costates.

Problem FRM$_P$ was fully solved to an NLP tolerance of 1×10^{-6} and a mesh tolerance of 1×10^{-4} using an initial guess produced by problem FR, and the altitude, angle of attack, and bank angle profile of the two are compared in Figure 35.

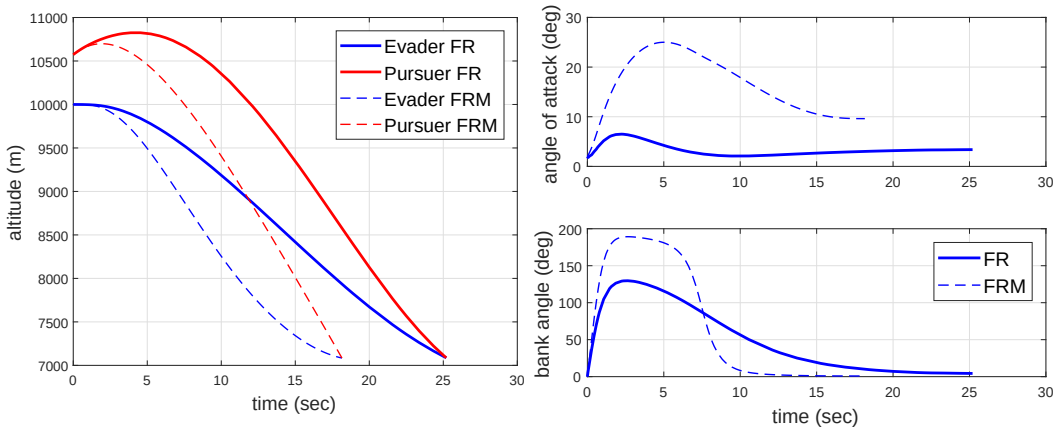

Figure 35. Altitude (left) and angle of attack and bank angle (right) of problems FR and FRM$_P$.

Not surprisingly, the two profiles are somewhat similar, and the solution to FRM$_P$ appears to be a reasonable candidate solution to problem FRM. Without checking that the necessary optimality conditions have been met, the user may assume that problem FRM$_P$ has in fact produced an optimal trajectory corresponding to FRM. However, it is possible to check this solution by investigating the final values of the x, y, and h costates. While the pursuer costates were directly calculated in FRM$_E$, the evader costates must be estimated from the states and Lagrange multipliers using Equation (2.15). The resulting values are shown in Table 5.

Table 5. Terminal costates for problem FRM$_P$, no transversality condition.

	λ_x	λ_y	λ_h
Evader	3.85×10^{-4}	3.10×10^{-4}	6.83×10^{-5}
Pursuer	-5.53×10^{-4}	-3.50×10^{-4}	1.45×10^{-4}

Clearly, the terminal costate conditions in Equation (5.27) have not been met. In fact, because the objective coded into the software is to minimize the final time,

the optimizer chose values of the evader's costates which would minimize the time to achieve intercept. This result should not be surprising, given that this is in fact the goal of the problem as it has been posed.

Another check can be made by solving problem FRM_E. Theoretically the two problems should yield the same solution, as was seen for the FXM problems. Problem FRM_E was solved to an NLP tolerance of 1×10^{-6} and a mesh tolerance of 1×10^{-4}, again using problem FR to estimate costates for the initial guess. Figure 36 shows the altitude, bank angle, and angle of attack compared to the solution for FR. This

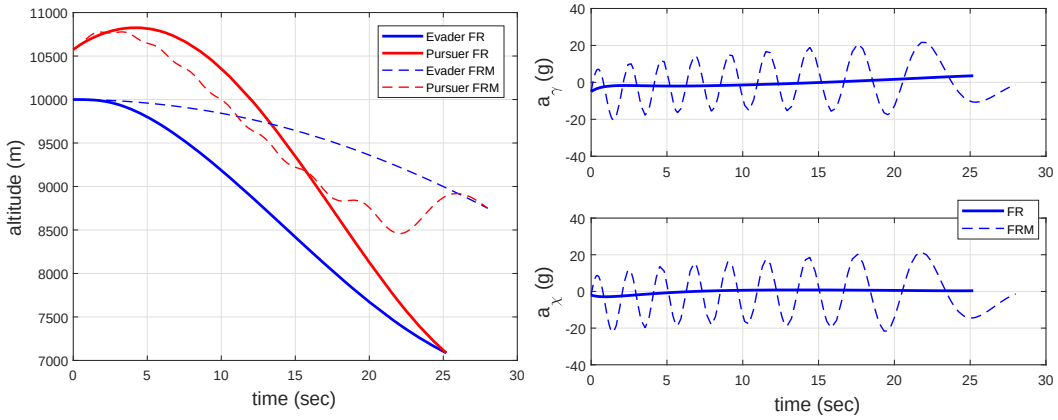

Figure 36. Altitude (left) and missile pitch and yaw accelerations (right) of problems FR and FRM_E.

trajectory is obviously different from the one found by solving FRM_P, and so unusual as to be obviously not the minimax. Essentially, the optimizer is directed to maximize t_f, but because pursuer costates are left free, values are chosen which prolong the flight as much as possible. In fact, for the shown trajectory, the final value for t_f was limited by the bound of 38 seconds put into the software. Without this bound, the optimizer could conceivably generate a trajectory which flies until the missile is nearly out of energy, before finally achieving intercept.

Problem FRM_P and FRM_E can be improved by applying the terminal costate constraints in Equation (5.27) to the final value of the Hamiltonian given by Equation (5.26), giving a new expression for the transversality condition which can be

used in FRM$_P$

$$\lambda_{x_E}(t_f)(\dot{x}_E(t_f) - \dot{x}_P(t_f)) + \lambda_{y_E}(t_f)(\dot{y}_E(t_f) - \dot{y}_P(t_f)) + \lambda_{h_E}(t_f)(\dot{h}_E(t_f) - \dot{h}_P(t_f)) = -1, \tag{5.30}$$

or likewise for FRM$_E$

$$\lambda_{x_P}(t_f)(\dot{x}_E(t_f) - \dot{x}_P(t_f)) + \lambda_{y_P}(t_f)(\dot{y}_E(t_f) - \dot{y}_P(t_f)) + \lambda_{h_P}(t_f)(\dot{h}_E(t_f) - \dot{h}_P(t_f)) = 1. \tag{5.31}$$

Problem FRM$_E$ has been solved, and is shown compared with FR in Figure 37. This time problem FRM$_E$ appears to be very similar to problem FR. However, a check

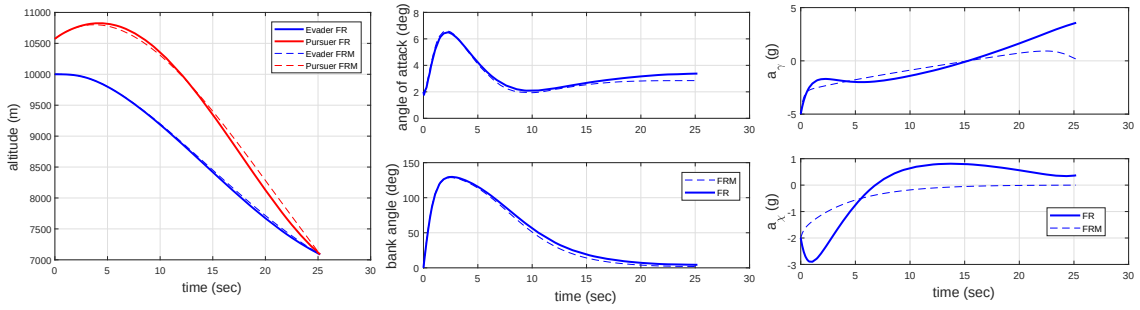

Figure 37. Altitude (left), angle of attack and bank angle (center), and pitch and yaw accelerations (right) of problems FR and FRM$_E$.

of terminal costates, shown in Table 6, reveals that the displayed trajectory it is not precisely the minimax trajectory defined by the necessary optimality conditions.

Table 6. Terminal costates for problem FRM$_E$, with transversality condition.

	λ_x	λ_y	λ_h
Evader	1.07×10^{-3}	6.68×10^{-4}	-2.16×10^{-4}
Pursuer	-1.07×10^{-3}	-6.67×10^{-4}	2.19×10^{-4}

As cited in Chapter II, the semi-DCNLP method has been used in several studies to solve relatively complex, free final time minimax problems. The authors define

an extended system to represent the semi-DCNLP method, along with necessary conditions for optimality as a post solution check. There is nothing preventing the method from obtaining a correct solution which satisfies the necessary conditions. However, as has been demonstrated here, there is no guarantee that a solution will satisfy these conditions for the current problem. Thus, while semi-DCNLP was a very useful technique for solving the fixed final time problems, the free final time solution must be obtained in another way.

5.3.3 Free Final Time Minimax Problem - Indirect Transcription.

The problem with semi-DCNLP for the FRM problem is that it is not possible to enforce the terminal costate conditions. This can theoretically be overcome by using the indirect transcription approach, where instead of only collocating costates for one player, both sets of costate dynamics and constraints are included in the software. This method, which simply seeks to solve the TPBVP via collocation, leaves the controls and objective empty (or with a value of zero) in the software since the necessary conditions account for both. Indirect transcription was shown to work for the FXM problem, although with an increase in computational effort.

All the necessary optimality conditions described for the FRM problem were implemented in the GPOPS-II software. The initial guess was created by solving problem FR and estimating the costates using Equation (2.15). Unfortunately, no solution to problem FRM was obtained using this method. While the error tolerances achieved by the NLP software were on the order of 1×10^{-5}, the problem always had at least one infeasible variable, usually being one of the terminal costate constraints. Inspection of the infeasible trajectories usually showed oscillations of the type seen in Figure 36. It is likely that the added transversality and terminal costate constraints make

the indirect transcription method too sensitive to achieve solutions for this specific problem. Thus a different solution method is required.

5.3.4 Free Final Time Minimax Problem - Decomposition.

The Decomposition iterative method, described in Chapter II of this document, may be used to solve problem FRM by forming two separate one sided optimal control problems. The method avoids the need to calculate the sensitive and unintuitive costate dynamics by iterating on these two problems. It is important to note that in the FRM problem, the evader and pursuer's objective and dynamics are completely separable, being linked together by only the terminal condition. Thus, rather than seeking to simultaneously solve the minimization and maximization problem, the Decomposition method finds the terminal condition which corresponds to the minimax by solving the dynamics and objectives of one player responding to the other, then repeating until convergence is achieved. The actual objective information is not shared between the two, instead the terminal position of each iteration is used.

The pursuer problem, here termed FRD_P, is to minimize the time required to reach the terminal position of the evader calculated from either the initial guess or the previous iteration. This point is termed $\mathbf{e}^i$ for short, and the capture condition is written as

$$\mathbf{\Psi} = (\mathbf{e}^i - \mathbf{x}_P(t_f))^T = \mathbf{0}, \tag{5.32}$$

where $\mathbf{x}_P$ is the terminal position of the pursuer at time t_f.

The evader problem, here termed FXD_E, in response to FRD_P, is to maximize the pursuer's minimized objective, or value, at the now fixed final time, t_f, obtained from the solution to FRD_P. Because an analytical expression cannot be written for the value of FRD_E, a linearized approximation is used, given by Equations (2.57) and (2.58).

For the current problem the Value of the evader simplifies to

$$V_E = \max_{\mathbf{u}_E} \mathbf{b}^T(\mathbf{x}_E - \mathbf{e}^i), \tag{5.33}$$

where $\mathbf{b}$ is the vector of Lagrange multipliers associated with the capture condition in problem FRD_P. The result of solving problem FR_E is a new final location for the evader, an updated value of $\mathbf{e}^{i+1}$. The algorithm described in Chapter II was used to iterate between solutions of FRD_P and FXD_E, beginning with an initial guess obtained by solving problem FR. Iterations of the two problems were continued until the square root of the sum of squares of the difference between the old and new values of $\mathbf{e}$ fell below 1×10^{-3} meters. Solving each subproblem to an NLP tolerance of 1×10^{-6} and a mesh tolerance of 1×10^{-4} required 6 iterations and approximately 70 seconds to converge.

It is possible to check the minimax solution by using the homotopy technique used in Chapter IV of progressively solving the fixed time problem for larger values of t_f, until the final range is nearly zero. In Chapter IV, problem FX approximated FR as the fixed t_f approached the free value where $r(t_f) = 0$. Now, problem FXM_E was solved for increasing values of t_f until the distance reached a value near zero. The intermediate fixed time trajectories displayed with the Decomposition solution in Figure 38 show that the solution to FXM progressively approaches FRM as t_f approaches the free final time value. This result builds confidence that the trajectory obtained using the Decomposition method is in fact a minimax. The homotopy method, while interesting as a check, is not actually a minimax because the fixed value of t_f can never quite reach the free value where $r(t_f) = 0$.

One benefit of using the Decomposition method is that it relies only on direct methods. This means that necessary optimality conditions are not required, eliminating the complicated costate dynamics and terminal conditions associated with the

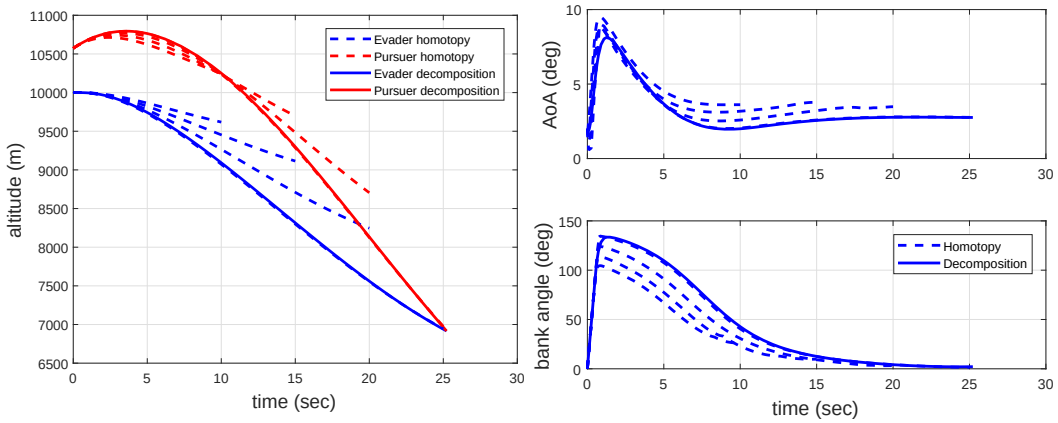

Figure 38. Altitude (left) and angle of attack and bank angle (right) found using the homotopy and Decomposition methods.

semi-DCNLP method. This is particularly helpful when state inequality constraints are present. For the fixed final time problem solved via semi-DCNLP, it was necessary to augment the objective with a penalty function to implement an altitude constraint. When using the Decomposition method, it is simply necessary to constrain the trajectories directly within the NLP. Problem FRM has been modified so the initial evader altitude is 1,000 meters, with the pursuer slightly above. The resulting minimax trajectory solved via Decomposition is shown in Figure 39.

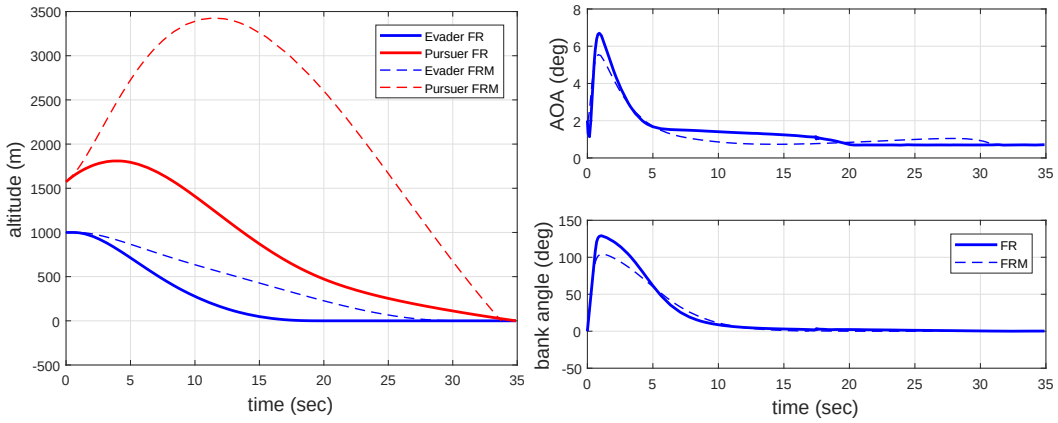

Figure 39. Altitude (left), angle of attack and bank angle (right) of the FRM trajectory solved with the Decomposition Method compared to the one-sided FR trajectory when altitude is constrained to be greater than zero.

5.4 Summary

The fixed final time minimax problem FXM is the two-sided analogue to the FX problem presented in Chapter IV. While the semi-DCNLP method offers a means of solving the FXM problem in a single NLP, it has a few drawbacks. First, formation of the costates and terminal conditions requires some amount of work to form the problem. This includes shaping the dynamics such that partial derivatives can be computed exactly. The presence of inequality constraints on the states or controls presents additional complexity, which here has been managed by transforming the control variable such that its range is bounded and by appending penalty constraints within the objective function. Finally, the sensitivity of the costates necessitates a very good initial guess, which can be generated relatively quickly using the results from the one-sided problem FX.

Unfortunately, problem FRM could not be accurately solved using the semi-DCNLP method. However, the Decomposition method, which requires iterative solutions of two one-sided problems, was implemented and successfully used to solve the FRM problem. While the Decomposition method may require more CPU time than semi-DCNLP, it can easily handle control and state inequality constraints without modification to the objective function.

While the minimax solutions themselves are interesting, their application to handling uncertainty is perhaps more enlightening. The definition of the minimax given by Equation (2.40) indicates that following the minimax trajectory serves as a guarantee of performance, no matter what control strategy the opponent chooses. By extending the definition of the control to other unknown parameters or states, the minimax can be applied to obtain a control strategy which accounts for the presence of uncertainty. This forms the basis of the next chapter.

VI. Pursuit-Evasion with Uncertainty

In Chapter IV, solutions to fixed final time free final state (FX) and free final time fixed final state (FR) optimal control problems were obtained under the assumption that perfect state information was available to the evader. The open-loop control was valid so long as the dynamic model, initial state, and control law of the pursuer were correct. Of course this scenario is impossible in practice, so techniques are required to deal with the reality of uncertainty. As this work has so far viewed problems from the evader's perspective, the following will focus on uncertainty in the state and behavior of the pursuer, although the same techniques could easily be applied to an uncertain evader.

Three scenarios will be examined in this chapter. First, the assumption of perfect information about the pursuer's state, model, and control strategy is no longer valid. However, the uncertainty is bounded by some limits, whether based on intelligence or by some previous observation. A structure is assumed for the uncertainty, and it is incorporated into the minimax problem. The solution will represent a lower bound guarantee on the performance of the evader, so long as the minimax control is employed.

The second scenario assumes that although the pursuer's state, model, and control strategy is uncertain, periodic state updates are available to the evader. The evader will then repeatedly solve for the optimal control after each update in order to correct its trajectory. This could be done using either the fixed final time (FX) problem as part of a Receding Horizon Controller, or the free final time (FR) problem as part of a Real-Time Optimal Controller. To evaluate the suitability of these two methods, two important factors are computation time and performance in achieving a satisfactory value of the objective. These factors will serve as the metrics to judge the suitability of

the RHC and RTOC methods for solving an optimal evasion problem in the presence of uncertainty.

Finally, to highlight the utility of the minimax trajectory generally, a minimum time to climb problem will be solved, but incorporating uncertainty using the minimax trajectory. This problem, which has energy elements similar to the evasion problems previously solved, is sensitive to uncertainty in the mass and thrust of the aircraft. Solving for the minimax will not only find a solution which incorporates this uncertainty, but it can also be used as a design tool, as will be demonstrated in this chapter.

6.1 No State Updates

6.1.1 Application of Problem FXM.

In the minimax solution obtained by solving problem FXM, it is assumed that both the evader and the pursuer have complete state information about each other, and will use this information to choose an optimal strategy. From the point of view of the evader, this problem is interesting to solve because it represents the worst-case scenario when the pursuer's guidance strategy is unknown. If the evader had perfect knowledge of the pursuer's dynamics and initial conditions, the open-loop control given by the solution to FXM would produce a lower bound for the objective, no matter what guidance strategy was used by the pursuer. Thus, following the minimax trajectory gives the evader a guaranteed minimum cost despite the uncertainty in the pursuer's control. The converse is true for the pursuer; the minimax control guarantees a maximum cost despite uncertainty about the evader's future actions.

To demonstrate this fact, the open-loop evader control produced by FXM_E has been used against a set of PN guided missiles with varying values of the guidance gain, N_P, between 2 and 5. All have the same initial conditions and are integrated

using Matlab's ode45 solver until the final time at 18 seconds. The scalar objective, the final distance $r(t_f)$, was recorded and is displayed in Figure 40 along with the scalar value found for FXM_E. As can be seen, none of the values of N_P used by the pursuer achieve a lower final distance than the minimax solution. This is because they are all suboptimal strategies, and thus subject to the minimax definition from Equation (2.40). Technically this relationship applies to the full objective, J, while only the scalar part has been plotted in Figure 40. However, the running costs are intentionally kept very small, and thus the guarantee may be considered to apply to the scalar cost, $r(t_f)$, alone.

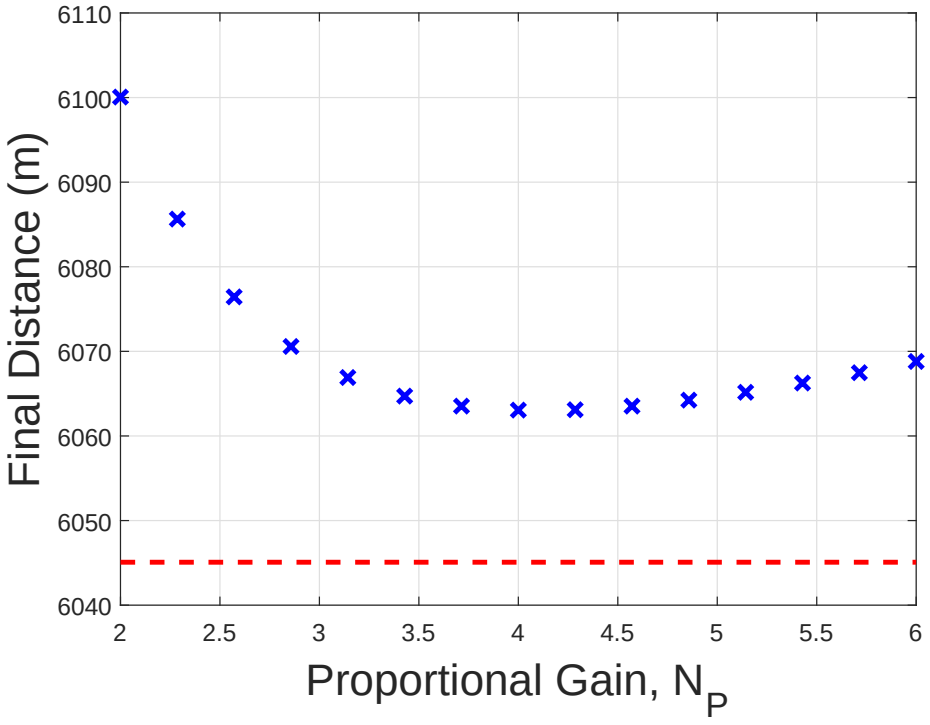

Figure 40. Final miss distance for a variety of PN missiles with gain N_P. The minimax value of the miss distance in dashed red is the theoretical minimum boundary.

To further demonstrate that the minimax control solution guarantees a lower bound for the evader, the initial bearing from the pursuer was varied, and problem FX was solved at $t_f = 18$ seconds for three levels of N_P. Problem FXM was also solved at the same final time, and the resulting final distance between the evader

and pursuer was recorded. The value of the FX distance minus the FXM distance has been plotted in Figure 41. The fact that the plotted profiles are always positive shows that the minimax value is the lower bound for a variety of problems, not just for one specific scenario. Interestingly, the $N_P = 2$ profile is nearly always 60 meters larger than the minimax. The $N_P = 4$ profile performs very well in a tail chase near 0 degrees bearing, but more poorly when a large turn is required. Figure 41 highlights how the minimax trajectory can serve as a benchmark for the relative optimality of other trajectories.

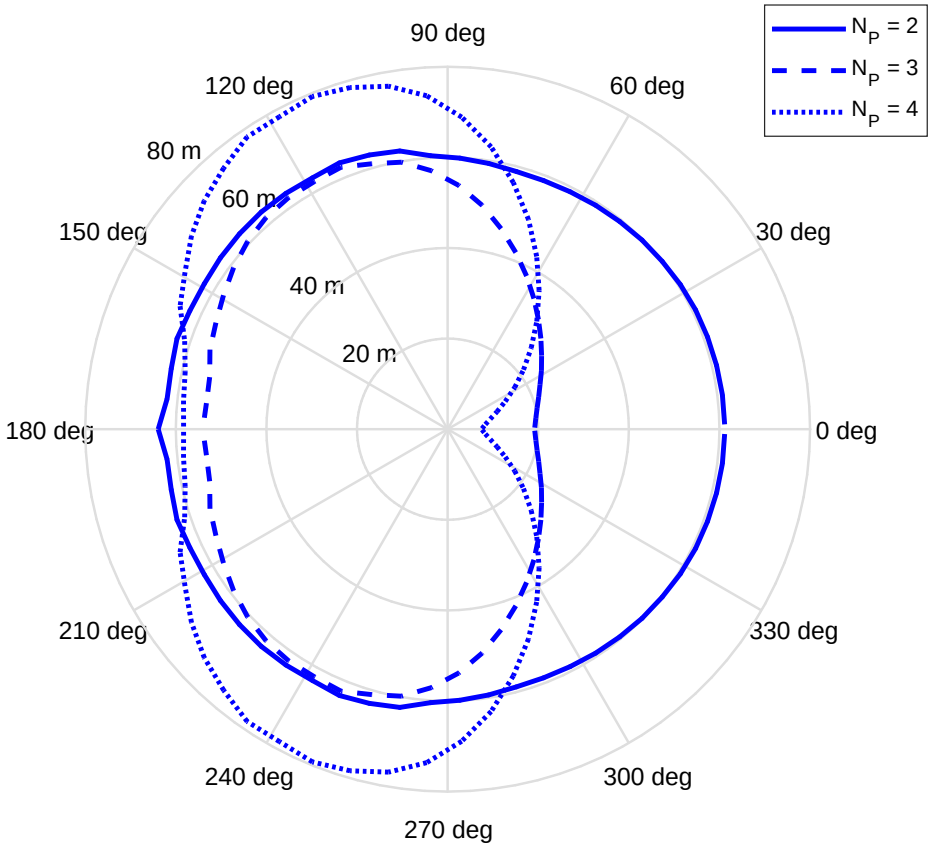

Figure 41. Final miss distance of problem FX minus the minimax value versus initial bearing angle with various N_P.

The concept of achieving a guaranteed cost despite uncertainty can be extended beyond the guidance strategy. For the unknown guidance strategy it was assumed that the pursuer could choose its control to minimize the objective function. Then

the pursuer was simulated as using a PN guidance strategy, and it was shown that no value of N_P could do any better than the minimax control. In a similar way, if there are unknown missile parameters, such as the navigation time constant, τ, the surface area of the missile, S_P, the mass of the missile, m_P, or the drag coefficients, $C_{D,0}$ and k, as defined in Capter III, it is possible to assume that these are control variables which the pursuer may choose. If problem FXM$_P$ is solved, these parameters can be coded as controls which will directly attempt to minimize the pursuer's objective. The resulting minimax trajectory will not only identify the open-loop controls of the pursuer and the evader, but also the values of the parameters which most benefit the pursuer, and the minimum value of the cost function against an optimal evader. The solution is once again a guaranteed lower bound on the cost.

As a demonstration, problem FXM$_P$ has been re-solved, but assuming that τ, S_P, m_P, $C_{D,0}$, and k may take values uniformly between 0.75 and 1.25 times their value shown in Table 3. These parameters were coded as bounded control variables in GPOPS-II, and the problem was re-solved. The minimax values of the parameters are mostly intuitive; m_P takes on the maximum value, while S_P, $C_{D,0}$, and k take on their minimum. To check against other values of the parameters that the minimax cost is once again the lower bound, a 5000 run Monte Carlo simulation was run, with values of the unknown parameters sampled from a uniform random distribution between the upper and lower uncertainty bounds. The open-loop evader and pursuer controls were used to propagate each simulated trajectory until the final time of 18 seconds. The resulting distance between the evader and the pursuer was recorded and plotted as a histogram in Figure 42, along with the minimax value of the scalar objective in dashed red. Clearly, the minimax is the lower bound, demonstrating the fact that for the evader the minimax control guarantees a minimum cost despite the uncertainty in either missile parameters or guidance strategy.

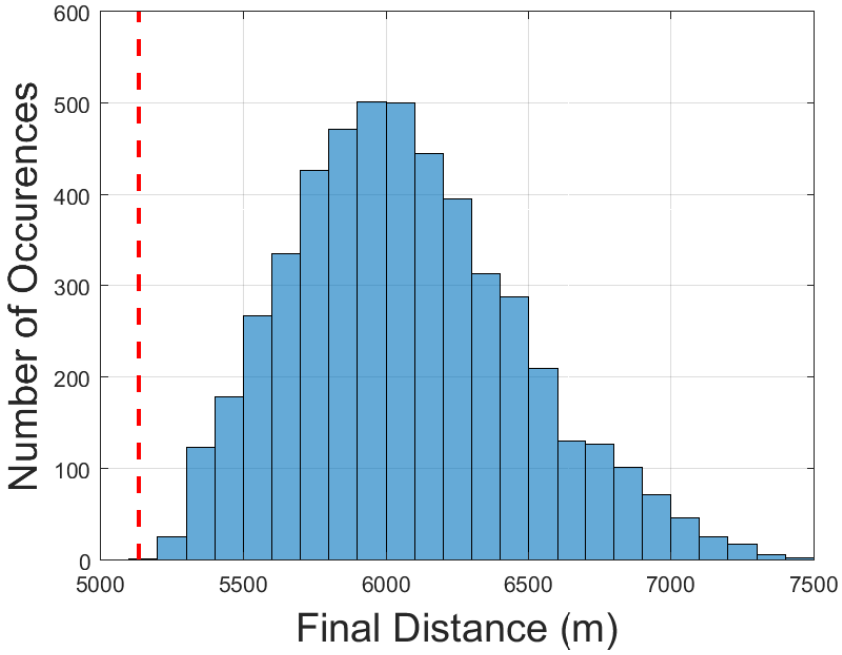

Figure 42. Histogram of the final miss distance for a variety of PN missiles with uncertain parameters. The minimax value of the miss distance in dashed red is the theoretical minimum boundary.

It is also possible to incorporate uncertain initial conditions in this way. For example, assume that exact values of the initial x_P, y_P, and h_P are unknown to the evader, but an estimate of their values is available, including mean and covariance, which assumes a multi-variate normal distribution. This is in fact a likely scenario given that if a launch is identified, the approximate last known location of the missile, including a mean and covariance, could be provided to the evader's guidance computer. In order to define the minimax trajectory, it is necessary to put a bound on the possible values of the initial states. This can be done through the Mahalanobis distance, given by

$$M = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})}. \tag{6.1}$$

For a multivariate normal distribution, the probability of a sample being within a given Mahalanobis distance is given by the chi-squared distribution [118]. For exam-

ple, if the covariance of the initial x_P, y_P, and h_P position is

$$\Sigma = \begin{bmatrix} 100^2 & 0 & 0 \\ 0 & 100^2 & 0 \\ 0 & 0 & 100^2 \end{bmatrix}, \tag{6.2}$$

then for a multivariate normal distribution, 90% of the points would fall within an ellipse of size M^2=6.25. This bound was implemented as a constraint on the allowable initial conditions within GPOPS-II, and problem FXM$_P$ was re-solved. In choosing its initial position, the pursuer improved upon its objective against the evader. The result shows the worst starting location against which the evader must react, and also the resulting final distance. If the evader used the minimax open-loop control against any other starting position, the resulting final distance would be higher than the minimax value. The initial pursuer position in the x-y plane and x-h plane are shown in Figure 43 with a large red dot.

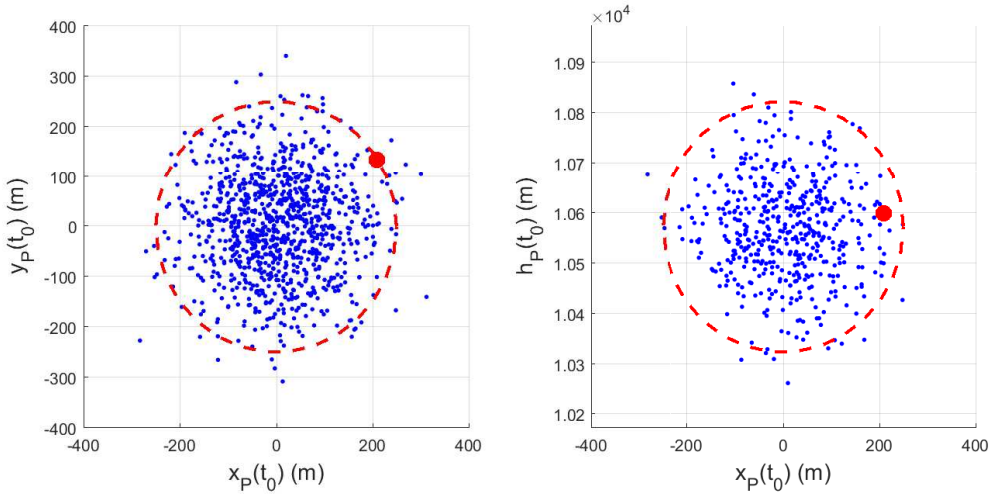

Figure 43. Starting positions for the Monte Carlo analysis of the FXM minimax problem with uncertain pursuer initial position. The minimax value is the large red dot.

A 5000 run Monte Carlo was run using the open-loop pursuer and evader controls, but with a multivariate normal distribution with mean from Table 4 and covariance

from Equation (6.2). The simulated pursuer initial positions are shown in Figure 43, along with the 90% boundary (Mahalanobis = 6.25). Note that some of the simulated starting points are outside the 90% boundary. For these points, the minimax guarantee does not apply. As might be expected, the minimax value of the pursuer's initial position lies on the 90% boundary nearest to the evader, to the upper right in the X-Y plane, while the minimax initial altitude is slightly above the mean value.

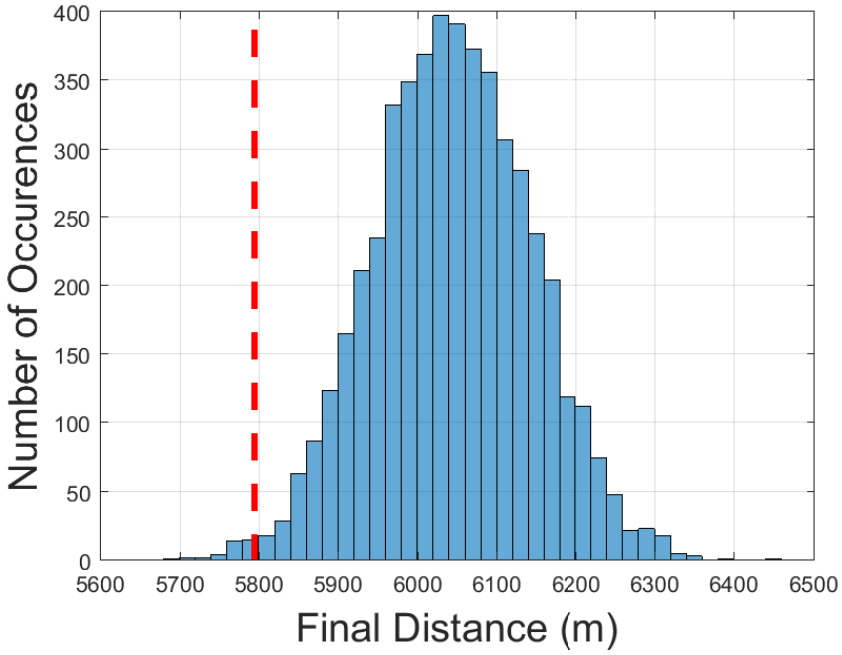

Figure 44. Histogram of the final miss distance for a variety of missiles with uncertain initial position. The minimax value of the miss distance in dashed red is the theoretical minimum boundary.

The final distances resulting from the Monte Carlo are shown as a histogram in Figure 44. It was guaranteed that 90% of the final distances would lie above the minimax value. As anticipated, some trajectories lie below this value, but 99% of the 5000 trajectories have a final distance higher than the minimax value.

When it comes to choosing an open-loop control during an actual missile evasion scenario, the minimax trajectory represented in Figures 43 and 44 is certainly a conservative choice. It is probable that by simply solving problem FX, a higher final

distance would be obtained. However, it is also possible that using the one-sided trajectory could result in even worse performance, depending on how far the actual missile is from the mathematical model.

6.1.2 Application of Problem FRM.

The solution to the FXM problem gives the evader a guaranteed lower bound on its objective at the fixed final time, no matter what control strategy the pursuer may use. This was demonstrated with a Monte Carlo analysis. In the case of FRM, a free final time problem with fixed final state, the lower bound guarantee is contingent on the terminal constraint and bounds of the problem. This means that applying the minimax control in a Monte Carlo simulation in open-loop is difficult, and unless performed correctly the results will not satisfy the minimax relation from Equation (2.40).

One issue with the FRM formulation is that it is impossible for any pursuer to numerically achieve $r(t_f) = 0$ meters exactly. While both the pursuer and evader have been modeled as point masses, it is reasonable instead to assign a capture distance of 20 meters (approximately the length of an F-4 Phantom). This means that the final constraint in both problem FR and FRM is $r(t_f) = 20$. In order to enforce this, an additional path constraint must be added, $r(t) >= 20$, to ensure the optimizer does not seek a solution where the missile passes through the evader to seek the constraint on the other side. A minimax solution to problem FRM with the new constraints is easily obtained using the Decomposition method as described in Chapter II and implemented in Chapter V.

It is guaranteed that if the evader uses the minimax control, the pursuer cannot achieve intercept at exactly $r(t_f) = 20$ any faster than the minimax final time, t_f. There are two likely results when the evader uses the minimax open-loop control

against any pursuer choosing a suboptimal control. The first is that the pursuer will leave the region where $V_C > 0$ earlier than t_f. The surface $V_C = 0$ is considered semi-permeable, meaning that the pursuer cannot re-enter the space of the game once it has departed. Thus the final intercept time would be infinity.

The second outcome is that the pursuer stays within $V_C > 0$, but does not achieve capture by t_f. After this time, the evader's open-loop control is no longer defined; however, the minimax time has already been exceeded so any behavior the evader chooses will result in a final time greater than the minimax value. This is the case when the minimax evader control is used in open-loop against a PN pursuer with $N_P \in [2, 5]$.

To demonstrate the second outcome, the value of N_P was varied between 2 and 5 in simulations where the evader control was the minimax, while the pursuer used PN. The distance between the evader and pursuer was recorded for each simulation, and is displayed in Figure 45. For all trajectories the closing velocity was positive until the final time, meaning that no instances of the first outcome were observed when using PN. As an aside, the benefit to the pursuer of choosing $N_P = 4$ is readily apparent.

6.2 Periodic State Updates

Modern aircraft may carry a missile approach warning system which could provide estimates on the state of an incoming threat. This information may be used to improve the open-loop solutions developed in the previous section. Each time an update becomes available, the optimal evasion trajectory may be re-solved, with the current time and state as the initial conditions. When the optimal evasion is obtained by solving a fixed final time problem such as FX, the ensemble of iterations is known as a Receding Horizon Control (RHC) problem. Posing the evasion as a free final

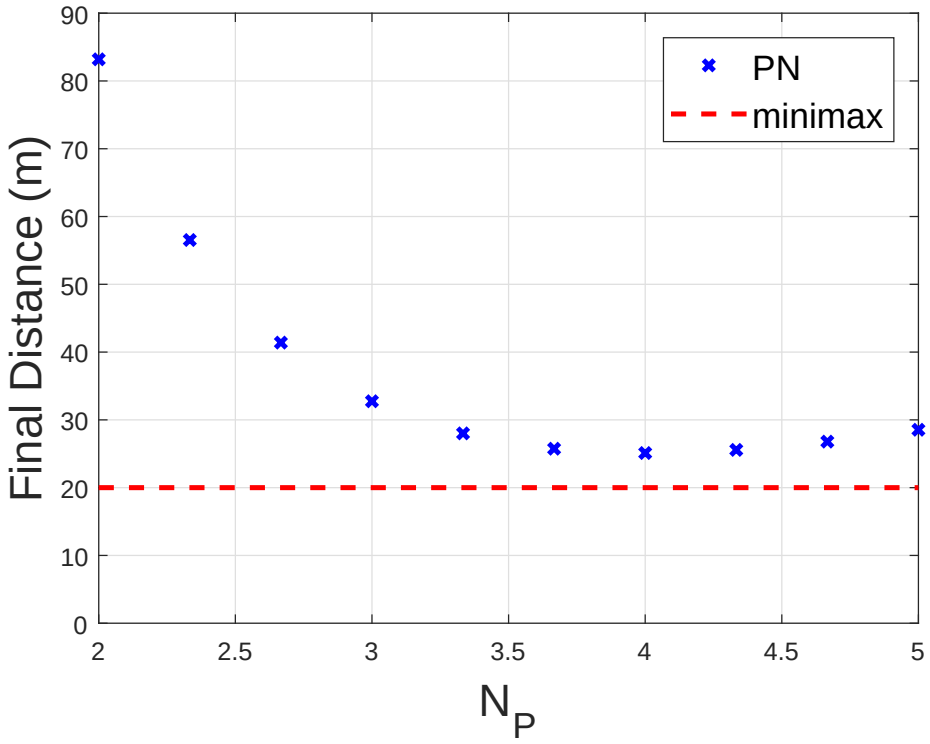

Figure 45. Distance between the evader and pursuer when the evader uses the minimax control from problem FRM, while the pursuer uses PN with various values of N_P.

time problem such as FR is called Real Time Optimal Control. Both these methods have been described in Chapter II, and will be used here—to investigate how state updates can be used to improve upon the result of an evasion scenario.

6.2.1 Receding Horizon Control.

In a typical application of RHC the optimal control problem is only solved for a short time span. However, in reference [44] it is shown that longer time horizon problems tend to perform better. In Chapter IV is was shown that the FX problem could replace the energy phase of an evasion trajectory up until approximately $t_{go} = 5$ seconds. Thus it makes sense to solve each iteration in the RHC problem with a fixed final time equal to $t_{go} = 5$ seconds. Unfortunately, t_{go} is not known at the problem start, and must be approximated. The simple estimate described by Equation (2.17) can be used when only range (r) and range rate (V_C) information are provided.

6.2.1.1 Correct model, perfect updates.

An example RHC problem has been solved by assuming that updates of the pursuer's state are available every second. For this example, it is assumed that the updates are perfect, without uncertainty. It is also assumed that each FX problem is solved instantly, and the solution is available at the exact time it was required. After each solution to problem FX is obtained, the optimal control is used open-loop in a simulation representing the true behavior of the missile and aircraft. For the example problem, the simulated pursuer is the same as the model used in problem FX. Given these assumptions it may be expected that the simulated trajectory should follow the optimal control solution exactly. This is not correct because problem FX is repeatedly calculated with a different value of t_f, since the t_{go} calculation incorrectly assumes that neither pursuer nor evader will maneuver.

With the initial conditions from Table 4, the first fixed final time corresponding to $t_{go} = 5$ seconds occurs at nearly $t = 16$ seconds. However, this is a conservative approximation. As each solution is calculated, the trajectory progresses, and the approximated time when $t_{go} = 5$ is pushed backward to 20 seconds when the evader begins the maneuverability phase. An overhead view and altitude profile of the FX trajectories compared to the simulated model is shown in Figure 46. The altitude profile shows that the first solutions to FX do not match the final trajectory due to the updated t_{go}.

6.2.1.2 Mesh Refinement.

An initial guess must be supplied for each attempted solution of problem FX. For the first iteration, the best guess can be given by integrating the equations of motion with a constant control. However, in subsequent iterations it makes sense to reapply the solution to the previous iteration as the initial guess. This homotopic

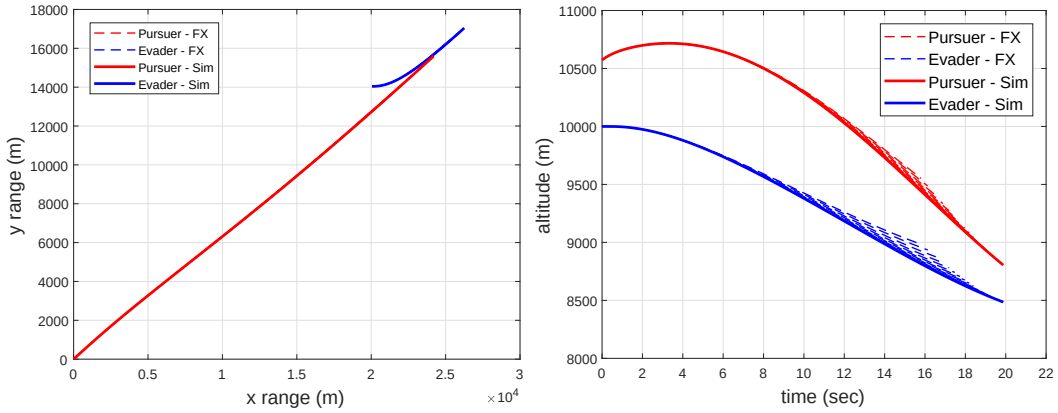

Figure 46. Overhead view (left) and altitude profile (right) of the RHC scenario solved using problem FX.

guess generation improves the convergence time dramatically, although it has the potential to trap the solution into a local minimum. A further improvement can be made by adapting the mesh to fit the next iteration's known time vector. For a PS mesh, this process is not trivial.

As detailed in Chapter II, a PS mesh is formed of several segments, each with a varying number of nodes distributed on the interval $\tau \in [-1, 1)$ based on the LGR formulation. During the solution of the first problem FX, this mesh is refined so the regions which have large state or control gradients are more densely populated with segments and nodes. This mesh structure not only improves the accuracy of the solution, but also enhances the speed of convergence. In order to conserve this structure the mesh must be modified at each iteration. An algorithm is proposed to accomplish this.

1. Retrieve the mesh from the solution to the previous iteration of the optimal control problem.

2. Define the cut time, t_{cut}, as the new initial time to problem FX.

3. Determine in which segment, i, the time, t_{cut}, resides.

4. Remove segments 1 through i.

5. Insert a new segment with a small number of nodes (4), beginning at t_{cut} and ending at segment $i + 1$.

6. If the final time has changed, trim or add segments as appropriate.

7. Renumber the segments and redefine all mesh related parameters.

8. Interpolate the states and controls onto the new mesh.

9. Estimate any integrals required in the initial guess as a ratio of the new and old mesh time durations.

An example iteration of this algorithm is shown in Figure 47. The previous solution's mesh begins at $t = 0$ seconds, and contains two relatively dense segments prior to the update at $t = 1$ second. To adjust the mesh, the first three segments are removed and replaced by a single segment with only four points, beginning at $t = 1$ second and ending at the next segment near $t \approx 1.35$ seconds.

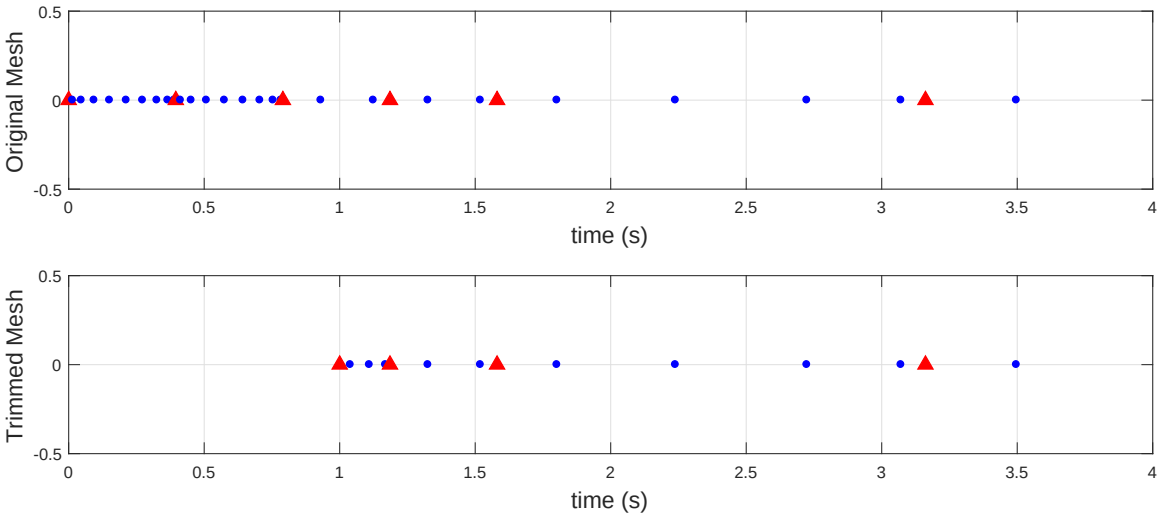

Figure 47. A single iteration of the mesh trimming algorithm. The update is received at 1 second, where the mesh is cut.

The RHC problem was solved with three different techniques for generating the initial guess at each iteration of problem FX. First, a single, unmodified guess was

supplied to all iterations. Second, the initial guess to each new iteration was taken directly from the solution of the previous iteration, including the mesh. Finally, the mesh from the previous iteration was adapted using the above algorithm. The resulting CPU time required to solve each iteration of problem FX was recorded and displayed in Figure 48. While the second method of recalculating the initial guess from the previous solution is a definite improvement over simply using the same initial guess, it still often requires a subsequent mesh refinement using a method such as [22] to reduce the discretization error to the required tolerance. However, the mesh adjustment algorithm, which may be considered a mesh pre-conditioner, usually results in a solution with no mesh refinement required.

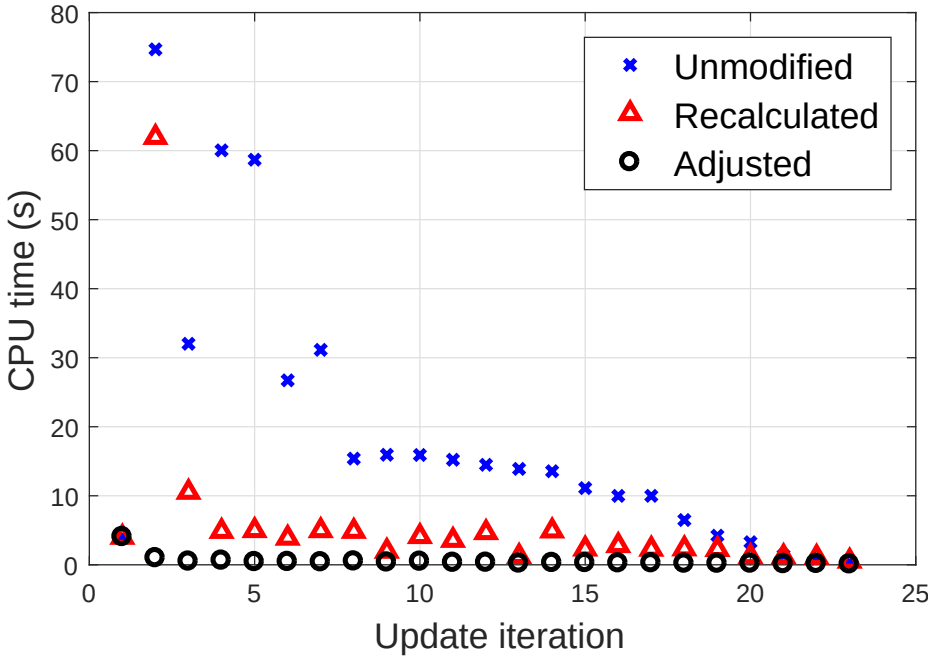

Figure 48. Comparison of computation times for iterations of the RHC problem using three methods of generating the initial guess.

Using the mesh trimming algorithm, the CPU time required to solve problem FX is typically around 0.25 seconds. In the GPOPS-II software, over half of this time is spent verifying the setup of the solution. If this process were removed, the CPU time

required to generate a trajectory update would be approximately 0.1 seconds, which is likely fast enough for real-time use.

Once the time corresponding to $t_{go} = 5$ seconds has been reached, it is left to the pilot to fly an evasive maneuver. This could be accomplished either by solving problem CPA or by simply executing a HGBR maneuver. Regardless, the updates are assumed to end at this point, defining the end of the RHC problem. The analogue to the objective from problem FX in the RHC problem is the distance between the evader and pursuer when $t_{go} = 5$ seconds. This final distance found using the idealized RHC solution shown above can serve as a benchmark as the unrealistic assumptions of perfect knowledge of the model and state are relaxed.

6.2.1.3 Incorrect Model.

Previously it was assumed that the pursuer used PN with $N_P = 4$. However, if this assumption is made but the actual simulated pursuer uses a different value of N_P, the performance of problem FX will degrade. For example, when no updates are available, if problem FX assumes $N_P = 4$, but in simulation $N_P = 2$, the pursuer will stray from the anticipated trajectory over time. Specifically problem FX will assume the pursuer uses much more turning acceleration than it actually does, with the result that the pursuer will not pull down or turn as quickly as expected, as displayed in Figure 49.

However, when updates are available the evader uses the pursuer's state information to modify its trajectory to compensate for the incorrect assumption on N_P. Figure 50 shows how the evader reduces its angle of attack and increases its bank angle to capitalize on the updates. The major effect is that the evader descends more quickly, further escaping the pursuer who descends less rapidly. This can be seen by close inspection of the altitude profiles in Figures 49 and 50.

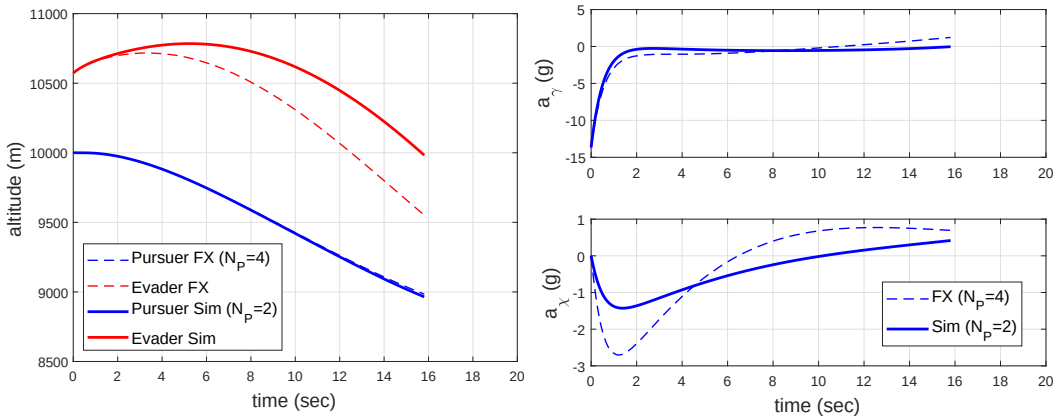

Figure 49. Altitude profile (left) and pursuer accelerations (right) with no updates solved using problem FX assuming $N_P = 4$ but in actual simulation $N_P = 2$.

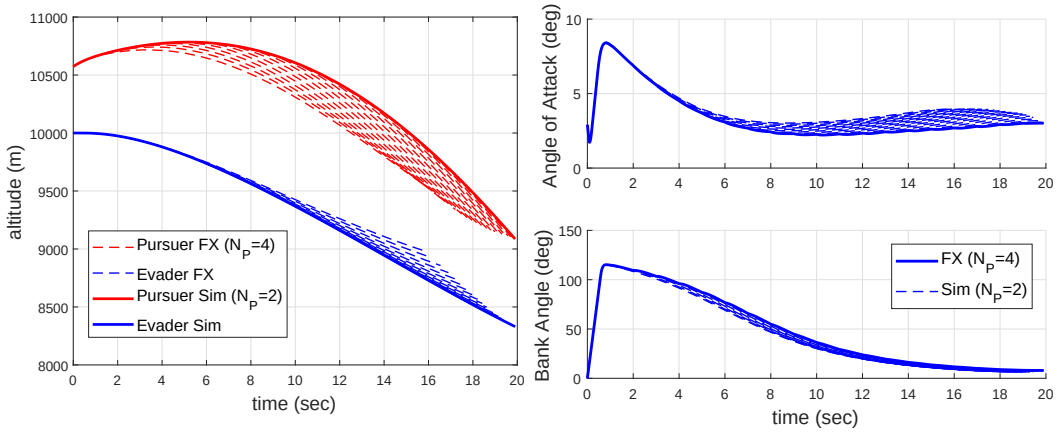

Figure 50. Altitude profile (left) and angle of attack and bank angle (right) of the RHC scenario with updates at 1 Hz solved using problem FX assuming $N_P = 4$ but in actual simulation $N_P = 2$.

When the t_{go} reaches 5 seconds, the iterations of the problem are completed. At this time, the CPA problem was initialized and run using $N_P = 2$ for the puruser, and the miss distance achieved was 83 meters. This is much higher than the miss distance values reported in Chapter IV, because the value of N_P is lower, resulting in a slower response time during the maneuverability phase.

The RHC algorithm corrects the evader's behavior when an incorrect assumption is made about the pursuer. This naturally leads to the question of whether the min-imax trajectory, solved via problem FXM$_E$, would represent a further improvement in the RHC technique. Calculation of each FX problem in the RHC can easily be ac-

complished within the 1 Hz cycle shown in Figure 50. Unfortunately the computation time of problem FXM_E is much too long for true real-time implementation.

6.2.1.4 Stochastic Updates.

In reality, any observation the evader makes of the pursuer would be subject to measurement noise. Here, for simplicity, it will be assumed that the noise on the pursuer's state is additive with a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \Sigma)$, where the covariance, Σ, varies linearly by the range between the pursuer and evader, and is given by

$$\Sigma = \frac{r(t)}{r(t_0)} \begin{bmatrix} \Sigma_1 & \mathbf{0} \\ \mathbf{0} & \Sigma_2 \end{bmatrix}, \tag{6.3}$$

where

$$\Sigma_1 = \begin{bmatrix} 100^2\ (m^2) & 0 & 0 & 0 \\ 0 & 100^2\ (m^2) & 0 & 0 \\ 0 & 0 & 100^2\ (m^2) & 0 \\ 0 & 0 & 0 & 30^2\ (m/s)^2 \end{bmatrix} \tag{6.4}$$

and

$$\Sigma_2 = \begin{bmatrix} 8^2\ (deg^2) & 0 & 0 & 0 \\ 0 & 8^2\ (deg^2) & 0 & 0 \\ 0 & 0 & (0.4g)^2\ (m/s^2)^2 & 0 \\ 0 & 0 & 0 & (0.4g)^2\ (m/s^2)^2 \end{bmatrix}. \tag{6.5}$$

In order to study the impact of receiving imperfect updates, 500 simulated RHC simulations were run. First, it was assumed that only the first noisy update, obtained by drawing from $\mathcal{N}$ was available at time zero. Problem FR was solved, and the evader's control was used against the pursuer in open-loop until the final time of

$t = 18$ seconds. The final range between the two was recorded. Next, for the same noisy initial conditions, it was assumed that updates were available at intervals of 3 seconds, obtained by sampling from $\mathcal{N}$ and additively applying the noise to the pursuer's current state. The final range was again recorded, and histograms of the two ranges are shown in Figure 51.

The mean of each distribution has been marked with a vertical dashed red line. While the mean final distance with updates appears to be larger than the case with no updates, this can not be demonstrated with the usual statistical tests because the distributions are non-normal. Instead, a two-sample Kolmogorov-Smirnov test can be used to compare the two distributions. Using the one-sided test, the null hypothesis is that the distribution with no updates is the same as the distribution with 3 second updates, while the alternative hypothesis is that the empirical cumulative distribution of the no update distribution is larger than the empirical cumulative distribution of the 3 second update distribution.

Using Matlab's *kstest2* function with a significance level of 0.01 on the data indicates that the null hypothesis should be rejected, meaning that the final distances with no updates tend to be smaller (a larger CDF) than those with 3 second updates. In other words, calculating updated trajectories statistically results in an improved final distance. This result is perhaps not surprising, although it is not a given that imperfect updates are better than none at all.

6.2.2 Real Time Optimal Control.

In RTOC, problem FR is repeatedly solved with new initial conditions each time a state update is available. Once again to begin it will be assumed that the update is noise free, and that the missile model in problem FR matches exactly with the simulated "truth" model. Unlike in RHC, this time t_f does not need to be predefined;

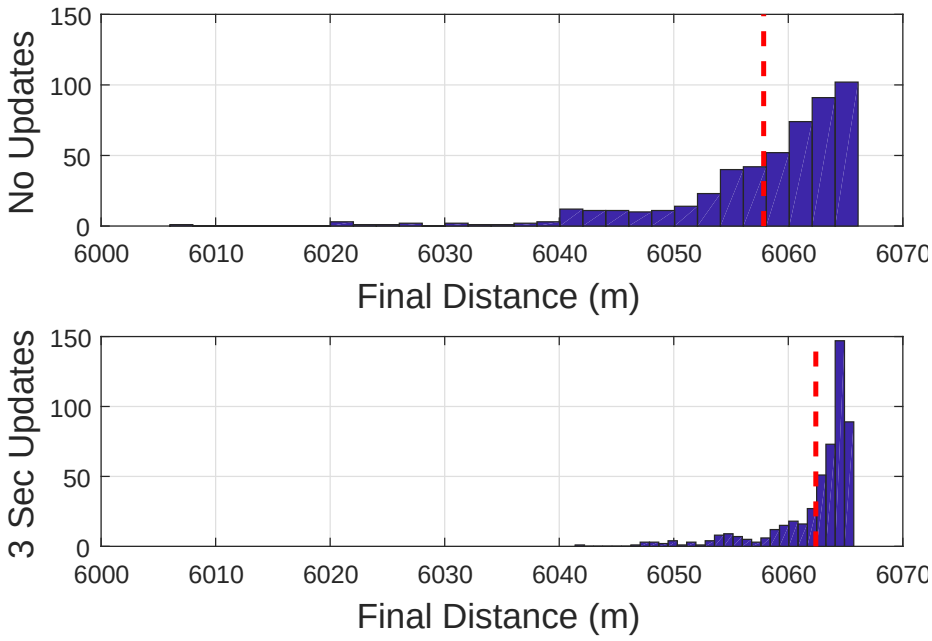

Figure 51. A comparison of the histograms for the noisy update RHC problem. The top distribution shows results obtained without updates, the bottom shows the results when updates are available every 3 seconds.

finding its value is the objective of problem FR. Still, t_{go} is needed to determine when the energy phase should end, and the maneuverability phase should begin. Once problem FR has been solved, a very accurate t_{go} estimate is obtained by simply subtracting the current simulation time from the t_f calculated in the FR trajectory. However, to better compare with the RHC method, the same estimate for t_{go} will be used, so that the final evasive maneuver will have approximately the same duration.

As a check, the RTOC problem was solved with an update frequency of 1 Hz. The resulting trajectory is shown in Figure 52. Note that unlike the comparable RHC trajectory displayed in Figure 46, the optimal control trajectories and the simulated trajectories match perfectly. This is the expected behavior which simply indicates that each FR problem has been solved to sufficient numerical accuracy that the integrated control over 25 seconds results in the same trajectory. This was not the case with the RHC trajectory due to the poor estimate of t_{go}.

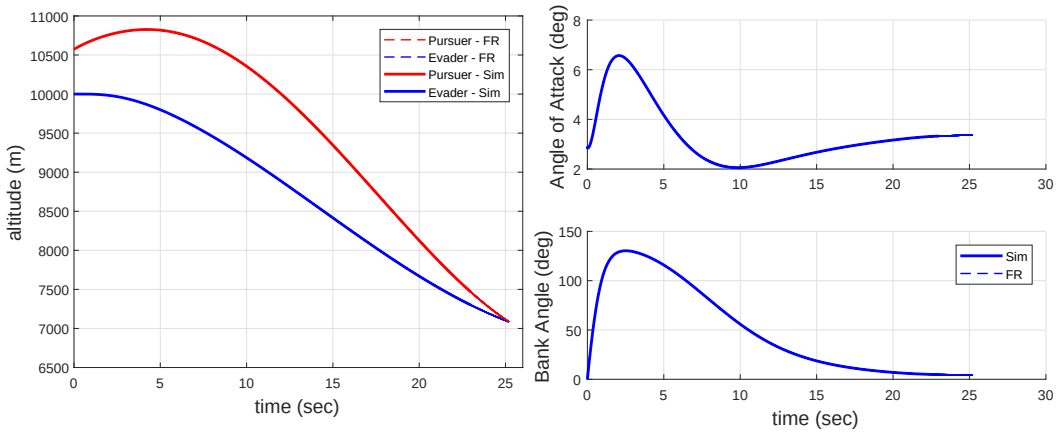

Figure 52. Altitude profile (left), angle of attack and bank angle (right) of the RTOC scenario solved using problem FR.

6.2.2.1 Mesh Refinement.

As in RHC, the mesh adjustment algorithm for the initial guess was studied. Three different methods of obtaining an initial guess prior to each iteration of problem FR were used and the resulting CPU times are compared in Figure 53. The first method was to simply use the same initial guess for all iterations. The second method was to use the solution from the previous iteration, including its mesh, as the guess for the next iteration. The third method was to apply the mesh adjustment algorithm presented with RHC. As can be seen, the mesh adjustment algorithm is once again an obvious improvement in the CPU time required to solve each iteration of problem FR. This also correlates to an improved likelihood of the problem actually converging.

As with RHC, the RTOC problem repeated solutions of the FR problem each time an update was received, until the simulation time reached $t_{go} = 5$ seconds. At this point the CPA problem was then solved to determine the best possible miss distance that could be achieved. The CPA trajectory again looks like a shortened version of the CPA solutions obtained in Chapter IV, being a well-timed roll at high g. The miss distance achieved for this idealized problem was 31.07 meters, consistent with the results obtained in Chapter IV, specifically as seen in Figure 27.

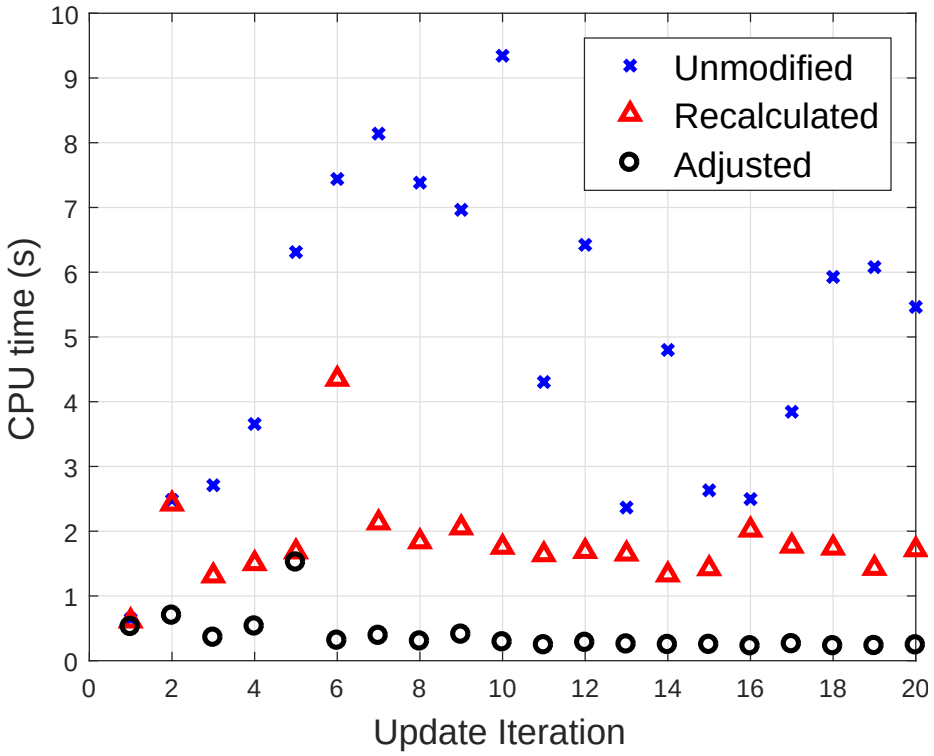

Figure 53. Comparison of computation times for iterations of the RTOC problem using three methods of generating the initial guess.

The previous solution was interesting only because it behaved as expected, that is the simulation trajectory matched the optimal control trajectory from problem FR exactly. If the simulated model of the pursuer does not match the model in FR, a different result is obtained. For example, if the pursuer in FR is assumed to use PN with a gain of $N_P = 4$, but in the truth model simulation it uses $N_P = 2$, the resulting trajectories obviously will not match. The FR solution and simulation trajectory are compared in Figure 54 for the scenario where no updates were received. Because problem FR expected the pursuer to use $N_P = 4$, it calculated a trajectory for the pursuer which dove too quickly, using a higher magnitude control effort. If the correct N_P of 2 were used, the final miss distance (after solving the CPA problem) would have been 134 meters. However, because problem FR expected that $N_P = 4$, it did not fully take advantage of the decreased control, and only achieved 111 meters.

Nevertheless, recall that the miss distance achieved by the RHC problem for this same case was only 83 meters, demonstrating once again the superiority of the FR problem over the FX problem in standing as a surrogate for the energy phase.

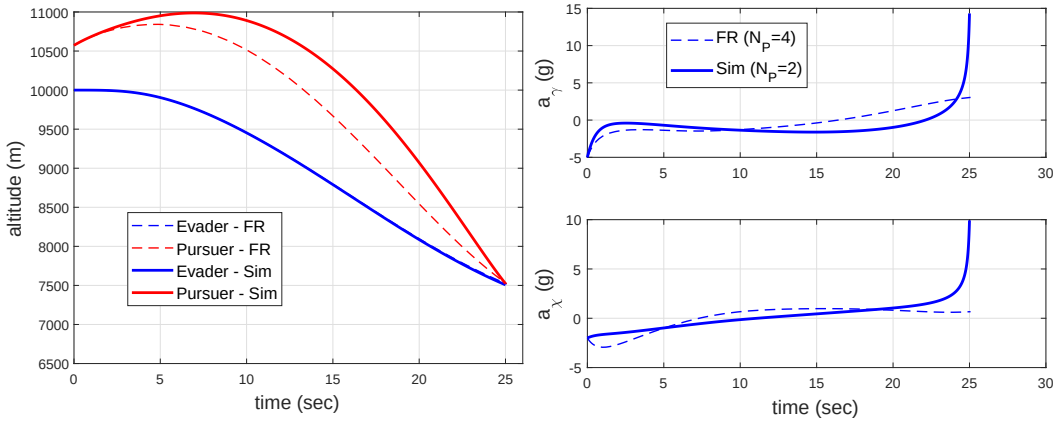

Figure 54. Altitude profile (left), longitudinal and lateral missile accelerations (right) when the missile gain is assumed to be $N_P = 4$ in problem FR when in simulation it is actually $N_P = 2$.

If updates are available, the situation can be improved for the evader somewhat. As the simulated trajectory strays from the anticipated FR trajectory, an update allows the evader to once again calculate an optimal trajectory. Figure 55 shows the repeated FR solutions compared with the actual simulated trajectory for an update frequency of 1 Hz, again using the incorrect value of N_P. The miss distance is increased to almost 130 meters.

6.2.2.2 Update Frequency.

The RTOC problem was solved with a variety of update frequencies and the miss distance at the end of the CPA problem was recorded and displayed in Figure 56. Note that the miss distance is very high because the simulated pursuer is using $N_P = 2$. When no updates are available, i.e. at an update rate of 0 Hz, there is no feedback that N_P is incorrect. However, as updates are generated, the optimal controller can somewhat compensate for the incorrect model with feedback. However, no matter how

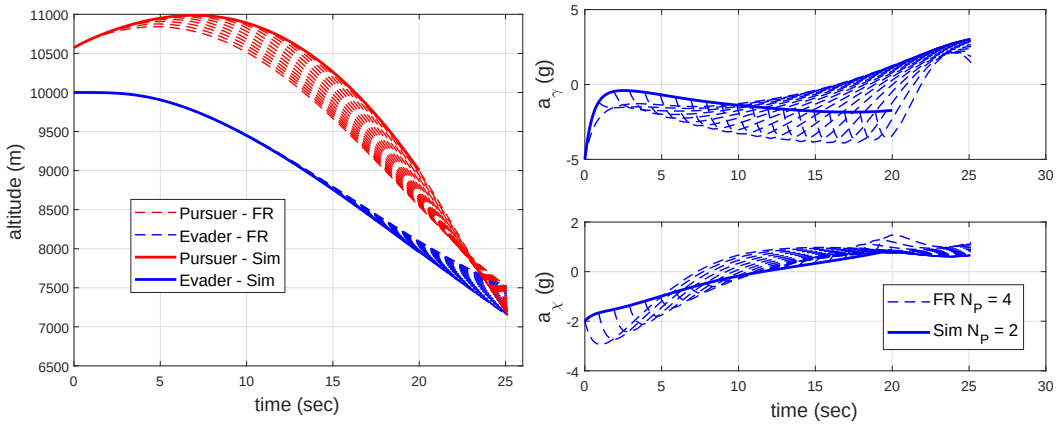

Figure 55. Altitude profile (left), longitudinal and lateral missile accelerations (right) when the missile gain is assumed to be $N_P = 4$ in problem FR when in simulation it is actually $N_P = 2$. Problem FR is updated at 1 Hz.

high the update frequency, the miss distance never reaches the theoretical optimal of 134 meters achievable if the N_P were chosen correctly. This demonstrates the intuitive result that repeated application of the wrong optimization problem is still suboptimal.

6.2.2.3 Stochastic Updates.

Further results are obtained if the assumption of noise-free updates is removed. The evader's estimate of the pursuer's state was corrupted by adding random noise from the multivariate normal distribution $\mathcal{N}(\mathbf{0}, \Sigma)$, where Σ reduces linearly with the separation distance and is given by Equation (6.3). A 200 run Monte-Carlo analysis has been performed to determine whether updating the trajectory with noisy updates serves to benefit the evader. In the RTOC case, the objective is to maximize the capture time, and this metric is compared for the case without updates against the case where updates are available every 3 seconds in Figure 57. Visually, it appears that the updates tend to shift the histogram slightly toward a longer capture time. Posing the null hypothesis that the two distributions are identical, against the alternative hypothesis that the cumulative distribution with no updates is larger than with up-

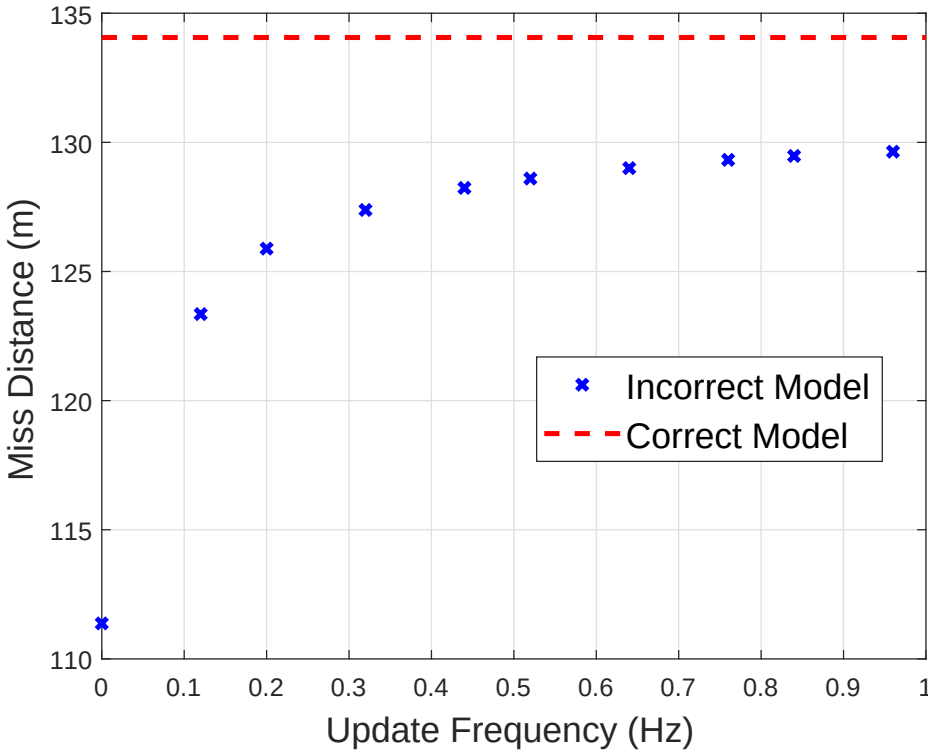

Figure 56. The effect of the update frequency on the miss distance in the RTOC problem with an assumed $N_P = 4$ when it is actually $N_P = 2$.

dates, a one-sided Kolmogorov-Smirnov test rejects the null hypothesis at the 1.0% significance level. Receiving updates does in fact tend to increase the capture time.

6.2.2.4 The Minimax in RTOC.

Although calculation of the minimax trajectory typically requires around 70 seconds, it is interesting to speculate how the minimax would perform when used for the evader's control in the RTOC problem. At each iteration, there is uncertainty about the actual position of the pursuer, and the results of the previous section showed that the minimax can be used to calculate a best response to the worst-case uncertainty. Ignoring then the large CPU time, the FRM problem has been used in place of the one-sided FR problem to calculate an updated trajectory every 3 seconds. a Monte

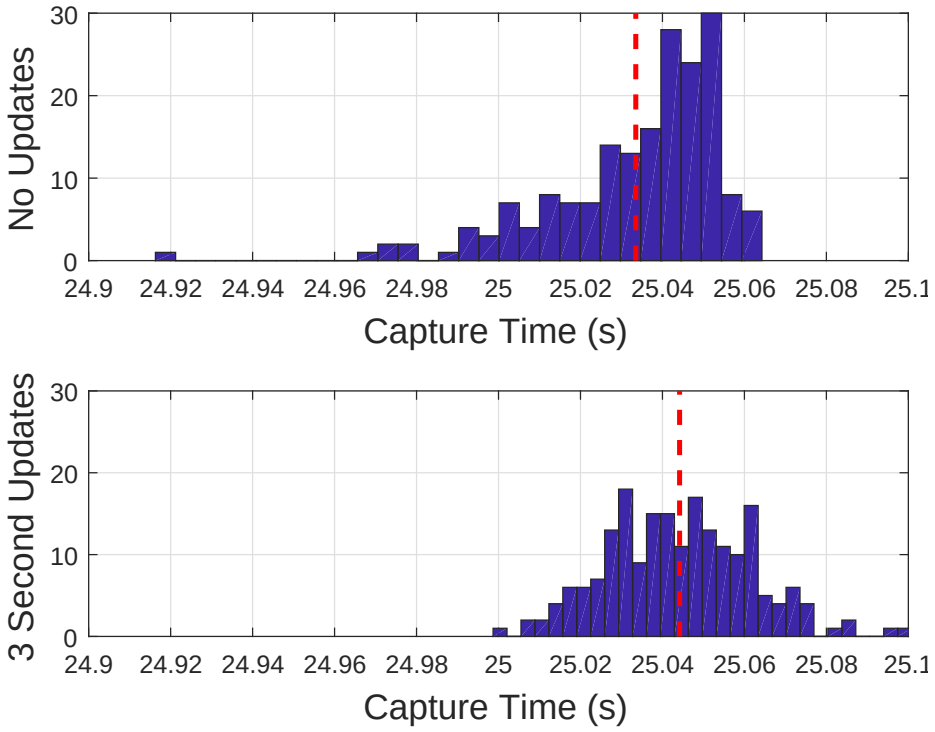

Figure 57. A comparison of the histograms of Capture Time for the noisy update RTOC problem. The top distribution shows results obtained without updates, the bottom shows the results when updates are available every 3 seconds.

Carlo of 150 runs was performed, and the capture time was recorded. The result has been displayed in Figure 58, compared to the result obtained using problem FR.

As might be expected, the FRM trajectory produces a lower capture time because it is a more conservative trajectory. Not only does it assume the pursuer may use an optimal control instead of PN, but it also assumes the worst-case state for the pursuer. Clearly the minimax trajectory has not proved beneficial in this case, although once again perhaps it is best to think of the minimax result as a benchmark against which other results may be compared.

6.2.2.5 The Miss Distance.

As seen in the previous figures showing histograms of the capture time, the differences between results are only spread over a few centiseconds. Although statistical

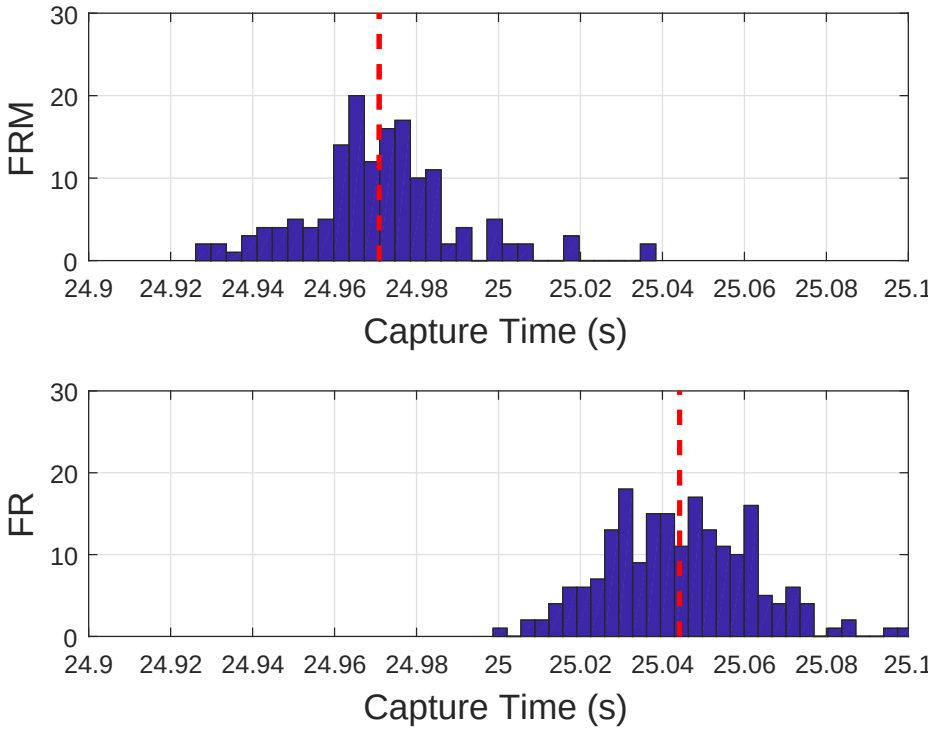

Figure 58. A comparison of the histograms of Capture Time for the noisy update RTOC problem. The top distribution shows results obtained using the minimax by solving problem FRM, the bottom distribution shows results obtained by solving problem FR.

tests may show a significant difference between two results at this resolution, the physical meaning of such a small time-span has not been demonstrated. Thus the sampled trajectories from the FR version of RTOC each were cut at 18 seconds, and the CPA problem was then run to calculate the best achievable miss distance. The data was recorded for the case without updates, and the case with updates every three seconds. The resulting histograms of the miss distance are displayed in Figure 59.

This time, as compared to Figure 57, the spread of the two histograms are more evidently distinct. Although no conclusion can be drawn about the mean value, it appears that the updates tend to narrow the variation in the miss distance. This is certainly beneficial for the evader, since it is less likely to achieve an unexpectedly low miss distance.

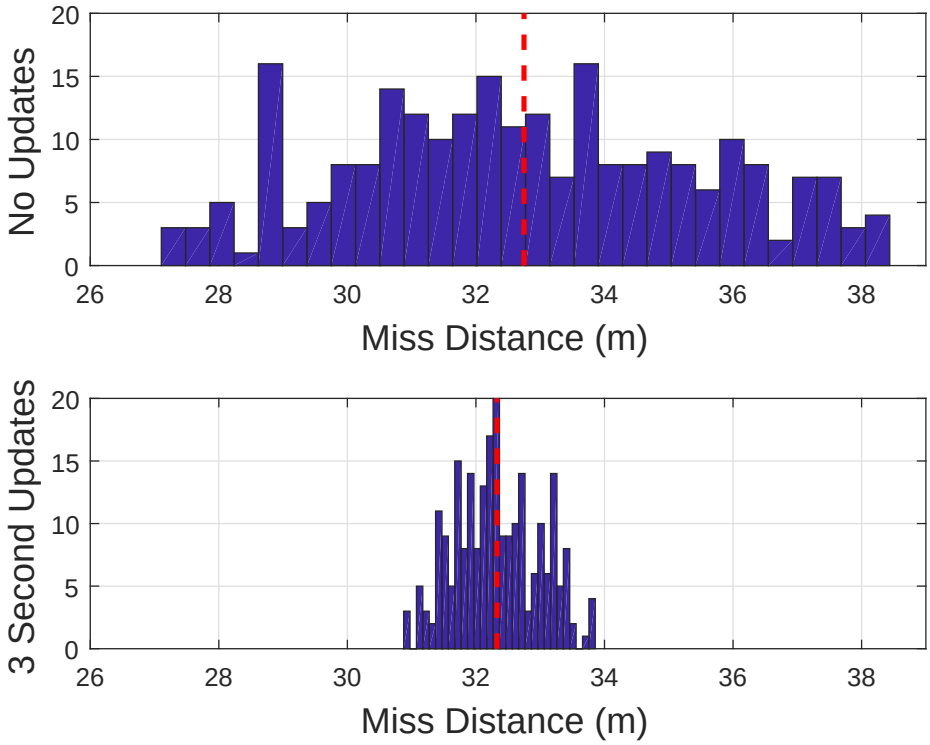

Figure 59. A comparison of the histograms of miss distance for the noisy update RTOC problem. The top distribution shows results obtained without updates, the bottom shows the results when updates are available every 3 seconds.

One final look at the Monte-Carlo data highlights an interesting relationship. Each of the samples was cut at the same time $t = 18$ seconds, following which the CPA problem was run to calculate the miss distance. At the moment when the trajectory was cut, the separation distance between the evader and the pursuer was recorded. The miss distance has been plotted against the separation distance at the cut time in Figure 60 to highlight a clear correlation between the two.

In the RTOC problem, the objective of each iteration is to maximize the capture time, t_f. This tends to put the evader further from the pursuer at the cut time of 18 seconds. This then is shown to correlate to a higher miss distance in the CPA problem. This correlation lends support to the idea presented throughout this work, that optimizing either the capture time or the separation distance during the energy phase will in fact serve to improve the final miss distance.

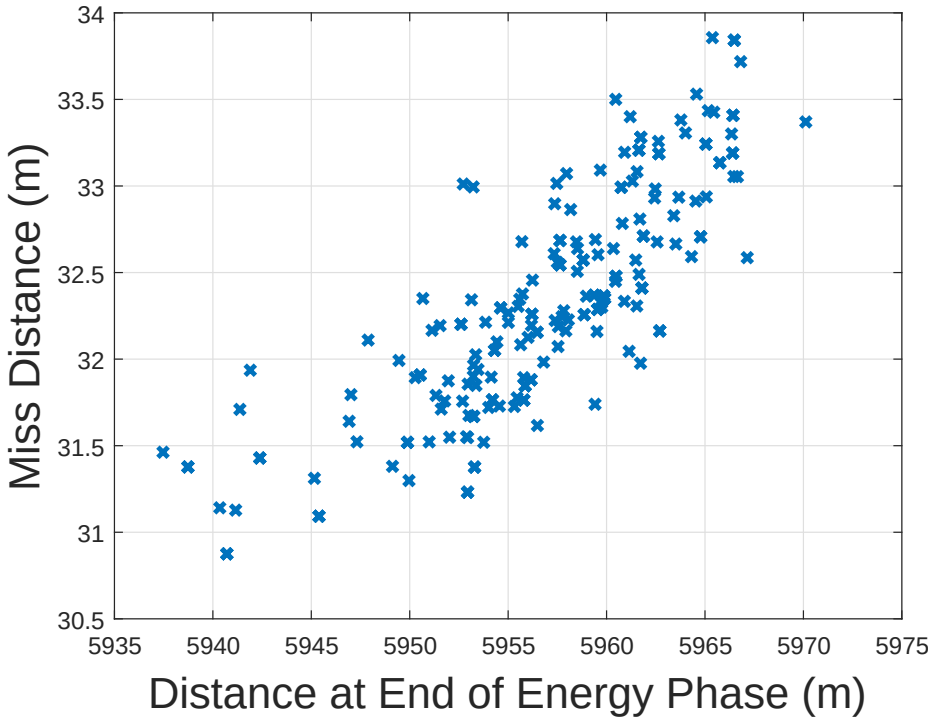

Figure 60. A plot of the resulting miss distance versus the separation when the RTOC trajectories are cut at 18 seconds.

6.3 Another Application of the Minimax

The miss distance achieved by the RHC and RTOC problems are encouraging, but for this problem the actual physical effect of uncertainty is not very large. For example, in Figure 51 the difference between receiving updates and not receiving updates is only about 0.1%. In Figure 56 the difference between having updates and not having them is only about 1%, and in Figure 60 the entire spread of the miss distances is only about 3 meters. It appears that the miss distance is not very sensitive to uncertainty. This is not the case for all problems, however.

A classic problem was proposed by Rutowski [64] wherein a supersonic aircraft attempted to reach a given altitude and velocity in minimum time. The surprising solution to this problem was for the aircraft to climb at maximum specific power until just prior to Mach 1, then perform a constant energy dive to supersonic velocity,

followed by another climb at maximum specific power until reaching the correct energy state, and finish with a constant energy climb to the desired altitude and velocity.

This problem, interesting from the point of view of energy exchange and thus relevant to the evader's energy phase, can be used to illustrate one final use of the minimax trajectory. The original authors solved the problem assuming that all aircraft parameters were known. However, supposing that some parameters were uncertain, the problem can be reposed as a minimax by imposing structure on these parameters, and assuming they are controlled by some adversary who wishes to maximize the time at which the aircraft reaches the final altitude and velocity. Thus the one-sided minimum time to climb problem becomes a two-sided minimum time to climb problem with uncertainty.

Suppose that the same aircraft defined by Equations (3.3) - (3.10) starts at zero altitude and M=0.38, and climbs in minimum time to M = 1.0 and 65,600 feet. The problem can be simplified by assuming that flight occurs in the vertical plane, thus removing terms involving y, χ, and μ. The costate dynamics and stationarity conditions have already been given in Equations (5.4) and (5.7), although they are similarly reduced to the vertical plane. The terminal costates are given by

$$\begin{aligned} \lambda_h(t_f) &= \mu_h \\ \lambda_V(t_f) &= \mu_V \\ \lambda_x(t_f) &= 0 = \lambda_\gamma(t_f) = \lambda_\alpha(t_f) = 0, \end{aligned} \tag{6.6}$$

and the transversality condition is

$$\lambda_h(t_f)V(t_f)\sin\gamma(t_f) + \frac{\lambda_V(t_f)}{m}(T\cos\alpha(t_f) - D) - g\sin\gamma(t_f)) + 1 = 0. \tag{6.7}$$

Although this problem can readily be solved using the PS method, making it unnecessary to re-solve the TPBVP, the method of indirect transcription can also be

used to solve the problem. Now, if the thrust and mass of the aircraft are known with $\pm$ 5% accuracy, it can be assumed that some "adversary" will use this uncertainty as a control to attempt to maximize the time to reach the desired final conditions. Thus the problem becomes zero-sum, and the minimax represents the best response of the aircraft to the adversarial uncertainty.

In order to solve this problem, the indirect transcription method can be coded into GPOPS-II, with the 5% uncertainty coded as the control, where the objective is to maximize the final time. Although the problem is free final time, fixed final state, unlike in the pursuit-evasion problem the number of unknowns and constraints are balanced for the semi-DCNLP method. Since the necessary conditions for minimizing the final time are already captured by the TPBVP, the resulting converged solution will be the minimax trajectory, which has been plotted alongside the deterministic trajectory in Figure 61. Of course the minimax results in a slower time, because it accounts for the worst-case uncertainty.

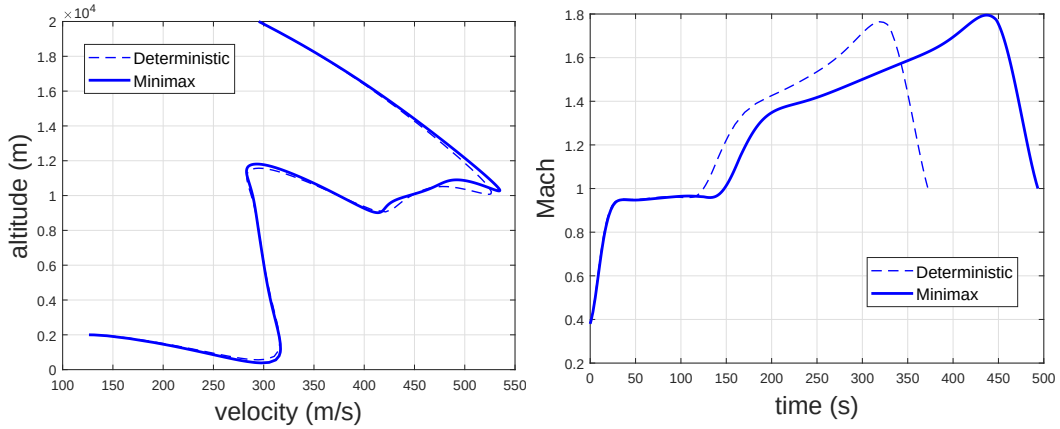

Figure 61. Altitude versus velocity (left), and Mach versus time (right) comparison of the deterministic and minimax trajectories.

To illustrate the usefulness of the minimax for this problem, suppose that two different aircraft designs are being compared. The first, Aircraft A, is as described in Chapter III, with maximum thrust given by Equation (3.10) and a constant mass

of 19050 kg. The second, Aircraft B, has 5% less thrust on average, but has a mass of only 16900 kg. The deterministic minimum time climb performance of the two aircraft are nearly identical, with each achieving the final altitude and velocity after approximately 375 seconds. However, when there is 5% uncertainty in the thrust and mass and the minimax is calculated, a large difference is seen in the final time, as shown in Figure 62.

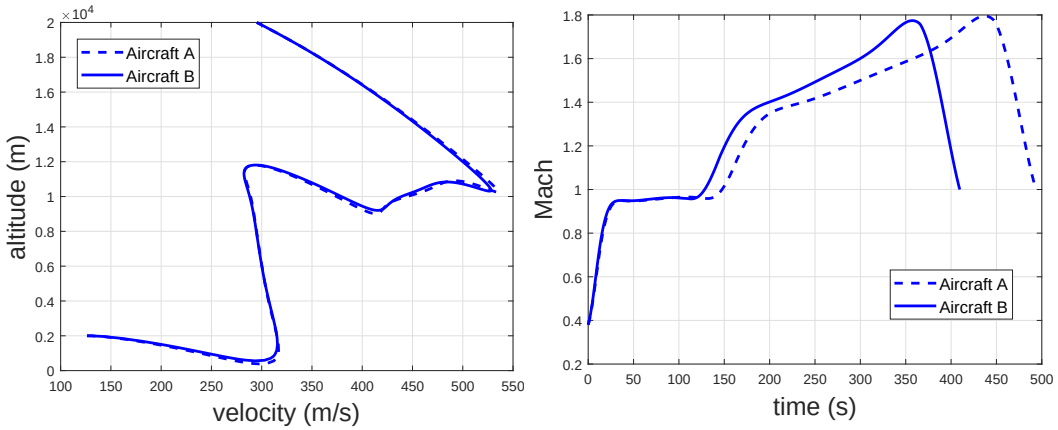

Figure 62. Altitude versus velocity (left), and Mach versus time (right) comparison of Aircraft A and B.

While the two aircraft have the same nominal performance, Aircraft B achieves a much lower time to climb, and is superior to Aircraft A when uncertainty is present. This was undetectable using the one-sided solution alone. Calculating the minimax trajectory is a quick and easy way to determine the robustness of a specific design to uncertainty.

6.4 Summary

The specific method used in handling a pursuit-evasion scenario depends on the information available. If the evader receives only imperfect initial state information about its adversary, the minimax solution may be solved to account for the uncertainty in the pursuer's initial conditions, the model, and the control strategy. The minimax

relation guarantees that the solution obtained will be the minimum so long as the evader uses the minimax strategy. Any deviation by the pursuer from the minimax will be of benefit to the evader.

However, if state updates are available, either RHC or RTOC may be used to improve upon the solution. The effectiveness of these methods has been demonstrated for problems where the model has been guessed incorrectly, and where the updated state information is noisy. While solutions cannot be obtained instantaneously, the FX and FR problems can usually be solved within a quarter second, meaning that real-time implementation of the two methods is feasible. It has been shown that receiving updates tends to improve upon the solutions, although the benefit may only be to reduce the uncertainty in the overall result. The RTOC problem was solved using the minimax solution, despite the large amount of CPU time required. No significant benefit of using the minimax with stochastic updates was detected.

Finally, a technique for incorporating uncertainty in a one-sided minimum time to climb problem was demonstrated by converting it into a minimax problem and using the unknown parameters as adversarial controls. Two different aircraft designs were compared and although no difference was found for the nominal case, the minimax solution showed a large difference in sensitivity to the uncertain parameters.

VII. Conclusions

7.1 Summary of Remarks

In this document, the ideas and concepts presented were intended to follow a logical pattern. First, after motivating and describing the problem in Chapter I, Chapter II provided a detailed description of the many numerical methods used in this work, primarily to give the reader an appreciation for the large body of research which exists covering the topics of missile guidance, energy-maneuverability, numerical optimal control, and differential game theory. Chapter III provided mathematical models of the aircraft and missile which would serve as a common base upon which all of the subsequent calculations would be made, with a particular focus on the energy exchange mechanisms, such as lift, drag, and thrust, of the models.

The research contributions began in Chapter IV by formulating the Closest Point of Approach evasion problem, which proved difficult to solve numerically because the terminal constraint in which the closing velocity must be zero exhibits a large gradient. This problem led to cutting the trajectory into two phases dominated by energy and maneuverability, respectfully.

Two different optimal control problems were proposed as surrogates of the energy phase, one posed as a fixed final time problem, FX, and one as a free final time problem, FR. Comparison of the two showed that while the FX problem was slightly more simple, the FR problem yielded better success in the subsequent maneuverability phase. It was demonstrated that problem FX became equivalent to problem FR as the fixed final time approached the capture time.

In Chapter V, two-player versions of problems FX and FR yielded minimax trajectories, which required special techniques to solve for complex dynamics. The semi-DCNLP method worked well for the fixed final time version, FXM, because the ter-

minal costate constraints were a function of the final states alone. Unfortunately the technique lacks closure for the free final time version of the problem, FRM. Instead, the Decomposition method was chosen to calculate FRM trajectories. As in the case for FX and FR, the FXM problem was shown to be equivalent to the FRM problem as the fixed final time approached the capture time.

The assumption that all information was available to both players was then relaxed in Chapter VI. When no state updates were expected, the minimax solutions to both fixed and free final time problems was used to capture the uncertainty in the model, control strategy, and initial conditions. It was shown that if the pursuer used a different control strategy from the minimax, its resulting objective was increased. Thus the minimax trajectory represented a guaranteed cost for the evader, so long as it used the minimax strategy.

However, if updates were available to the evader along the course of the trajectory, the information could be used to improve the result. The fixed final time problem was used as part of a Receding Horizon Control scheme, while the free final time problem fit into the Real Time Optimal Control method. Each of these was demonstrated, including scenarios where the control strategy of the pursuer was incorrect, or the state updates were stochastic. It was seen that by using grid adaptation, solutions could be obtained quickly enough for real-time implementation. Results of Monte-Carlo simulations showed that when trajectory updates were calculated, the uncertainty in the resulting miss distance was generally reduced.

Finally, the utility of the minimax was demonstrated for capturing uncertainty even in one-sided problems by solving a minimum time to climb problem. Two designs were compared which for the nominal scenario yielded nearly equivalent results. However, the minimax solutions showed that one design was much less sensitive to

variations in uncertain parameters. Thus the minimax can be used as a benchmark for making aircraft performance design decisions despite the presence of uncertainty.

7.2 Contributions

Within the summary of results described above, a number of novel concepts were mentioned. They represent the research contributions to the general body of missile evasion and are listed here.

1. Demonstrated that the fixed final time problem becomes the free final time problem as the fixed final time approaches the capture time (Chapter IV, Section 4.3).

2. Outlined a procedure to obtain an initial guess of the costates to use in the semi-DCNLP method (Chapter V, Section 5.2.2)

3. Proposed a penalty function for semi-DCNLP problems with a pure state constraint (Chapter V, Section 5.2.4).

4. Described an issue with using semi-DCNLP on certain free final time problems, and demonstrated the Decomposition method as an alternative solution technique (Chapter V, Section 5.3).

5. Demonstrated how the minimax solution represents a guarantee on the evader's performance despite uncertainty in the pursuer's model, guidance law, or initial state (Chapter VI, Section 6.1).

6. Developed an algorithm to improve the computational speed of complex RHC and RTOC problems by adjusting the mesh between each iterated solution (Chapter VI, Section 6.2.1.2).

7.3 Future Research

The speed of the PS method applied to this problem has shown that it could theoretically be used in real time aboard an aircraft. While the GPOPS-II software requires Matlab and is thus difficult to deploy on small computers, an open source optimal control code called PSOPT [19] was successfully compiled onto a Raspberry Pi 3 and an ODROID-C board. This may be an interesting direction for future study in pursuit-evasion. The details of this installation and testing are contained in Appendix C.

It was shown in Chapter VI that by adjusting the PS mesh, solutions could be obtained very rapidly. Both methods of calculating the minimax required longer computation times, enough to eliminate the possibility of using the minimax in real-time. It is likely that further work in adapting the initial mesh will result in improved convergence times. Two algorithms for mesh adaptation developed during the course of this work are described in detail in Appendix A and Appendix B.

It was also shown that while increasing the frequency of updates resulted in an improved solution despite uncertainty in the model, the improvement was limited by the fact that an incorrect model was being repeatedly used. A model identification step could be included in either the RHC or RTOC method each time an update is received. This could even be done using the PS method by attempting to solve for a set of parameters which might best match the estimated trajectory. Little or no mesh adaptation would be required if the system identification problem were set up on the same mesh as the previous solution to the optimal control problem.

In this work the evader and defender were alone. It is highly probable that multiple pursuer missiles would be encountered in realistic scenarios. Additionally, there may be friendly aircraft in the vicinity which could aid the evader in some way, and finally, the evader may help itself by launching a counter attacking missile. These

are all still technically two-team games, for which the techniques highlighted within this work would still apply. Also, no attempt has been made here to incorporate the use of other counter-measures. Solving optimal control problems which maximize the effectiveness of counter-measures is likely to produce interesting results.

As admitted in Chapter I, the scope of this work was limited to software only. However, as demonstrated in Chapter VI, the algorithm can be made to execute quickly enough for real-time implementation. Adaptation of pursuit-evasion algorithms onto hardware opens up several possibilities. The most obvious is to use these methods on a fully autonomous system. The dynamics and constraints would be adjusted to match the capabilities of the specific aircraft, and known threat models could be loaded at run-time.

Another option is to use the software as a suggestion service to manned pilots. When a threat is detected, the pilot would be given the option to allow the aircraft to autonomously fly an energy profile, leaving the pilot several seconds to worry about other problems, such as applying counter-measures. When the t_{go} neared 5 seconds, the pilot could be given the prompt to retake control of the aircraft and perform evasive maneuvers, or to again allow the aircraft to perform an autonomous maneuver such as the HGBR, S-turn, or a linear optimal solution. This would still allow a human pilot to manage the evasion encounter, while allowing them to focus on other tasks.

VIII. Appendix A

The following document was prepared for submission to the American Institute of Aeronautics and Astronautics Journal of Guidance, Control, and Dynamics in 2016. The document was not accepted for publication due to lack of interest from the reviewers, so it will be archived here.

A Mesh Adaptation Scheme for Path Constraints in Direct Collocation Optimal Control*

Ryan W. Carr†, and Richard G. Cobb‡

Air Force Institute of Technology, Wright-Patterson Air Force Base, OH 45433

Nomenclature

$\mathbf{B}$	Boundary conditions (event constraints) vector
$\mathbf{C}$	Path constraints vector
$\mathbf{D}$	The differentiation matrix
D	Aerodynamic drag (N)
J	Cost functional
L	Aerodynamic lift (N)
$P_{\dot{C}}$	Lagrange polynomial interpolation approximation of the rate of change of the path constraint
Q	The number of new collocation points to add to an interval
q	Aerodynamic heating along the leading edge of the wing, (BTU/ft^s/sec)
r	Radius from Earth's center in kilometers
r_{ko}	Keep-out zone radius in kilometers
t	Time in seconds
$\mathbf{u}$	Control variable vector
V	Vehicle velocity in kilometers per second
$\mathbf{x}$	State variable vector
x	Vehicle coordinate in kilometers
x_i	i'th discretized value of the x coordinate
x_{ko}	Keep-out zone x coordinate
y	Vehicle coordinate in kilometers
y_{ko}	Keep-out zone y coordinate in kilometers
γ	Flight path angle in radians
Φ	Mayer portion of the cost functional
ϕ	longitude in radians
ψ	vehicle heading in radians
τ	Collocated time
θ	latitude in radians
$\mathscr{L}$	Lagrangian portion of the cost functional

I. Introduction

The field of optimal control has recently benefited from the emergence of direct collocation methods, where the continuous state and control are discretized at a specific set of points allowing the dynamics to be transcribed to a static nonlinear programming (NLP) problem. The problem can then be solved by existing NLP software such as SNOPT[1] or MATLAB's *fmincon*.[2] If the collocation points are chosen to be the roots

*Distribution A: Cleared for public release, case number 88ABW-2015-6194

†PhD Student, Department of Aeronautical and Astronautical Engineering, AIAA Member.

‡Professor, Department of Aeronautical Engineering, Associate Fellow AIAA.

of an orthogonal polynomial, such as in Gaussian Quadrature, this method may converge exponentially and is often called pseudospectral, although the authors prefer to call the technique Direct Orthogonal Collocation.

The NLP solver attempts to minimize the objective function subject to a set of constraints. In direct collocation, the dynamics are enforced by equality constraints which must be satisfied within a desired tolerance at the collocation points. This does not, however, give any guarantee that the dynamics are feasible between points. Because the discretization is performed using a Lagrange interpolating polynomial, the error between points is bounded by the Cauchy interpolation error theorem.[3] It is possible to reduce the error bounds between these points by choosing an appropriate spacing for the points, such as by using the roots of a Legendre polynomial (hence the relation to Gaussian Quadrature), and also by increasing the number of points used in the interpolation. Failure to perform some kind of refinement on the point spacing can result in very large errors, even for a fully converged NLP solution.[4]

Many publicly available optimal control software packages implement such mesh adaptation algorithms.[5–9] Several schemes exist for adapting the point placement, or mesh, in order to reduce the error between points once the NLP has converged to a solution for the transcribed problem. In direct orthogonal collocation, these methods involve either increasing the number of total points and hence the order of the interpolating polynomial and regenerating the point spacing based on Gauss Quadrature (p-method), or dividing the solution space into distinct intervals (h-method). The two intervals are then linked by additional constraints in the NLP. The h-method results in an a piece-wise polynomial approximation of the state and controls.

One adaptation algorithm uses the differentiation matrix, D, to calculate the time derivative of the control. At a point where the control derivative is larger than some threshold value, a "knot" is placed to divide the mesh into two intervals.[10] Another author suggests a ph-method in which the error is defined by the difference between the interpolated state and an integrated approximation. The number of points is first augmented, and if the polynomial exceeds some limit, the interval is divided.[11]

Another scheme calculates the error in the dynamics at the midpoint between two points using the differentiation matrix, then either splits the interval or adds a fixed number of points depending on whether the error detected exceeds a local or global threshold.[12] Another method uses a similar error, but evaluated at a large number L of uniformly spaced points interpolated over the mesh interval (e.g. L=1000). If the error exceeds a threshold the curvature of the state or control is calculated using the first and second derivatives. If the curvature value is too high, the interval is divided. Otherwise the polynomial order is increased.[13]

II. Motivation

The mesh adaptation methods described above are shown to reduce error and improve convergence in the solution of a generalized optimal control problem. In some problems, a path constraint is used to further bound the feasible space of the state and control within the time interval of interest. Beyond simply limiting the states to an upper and lower limit, a path constraint may evolve through the trajectory, or involve complex interactions between states and controls. For example, the dynamics of a hypersonic glide vehicle may not require g-load, dynamic pressure, or friction heating as a specific state, but these quantities may be derived from a combination of the states and controls. It may be necessary to limit one or all of these along the trajectory using path constraints. The upper and lower bounds of a path constraint can be provided to the NLP as additional constraints at each collocation point.

When a path constraint is implemented in the NLP, the value of the path is restricted only at each collocation point. When a converged solution is reported by the NLP, the value of the path at these points satisfies the upper and lower bounds set by the user. However, in between collocation points it is possible that the path constraint is violated. Reference[12] mentions checking this value at the midpoint between two collocation points, but this still leaves the half of the distance between collocation points where the path constraint may be violated. Reference[13] specifically addresses problems where the path constraint is active, but because the method relies on checking the curvature, requiring estimation of the second derivative, it "creates a great deal of noise in the error estimate ...", making it "computationally intractable when a high-accuracy solution is desired."[6] In addition, the number of points where the error is evaluated is chosen to be a high number, 1000, but this seems arbitrarily high, akin to using an exhaustive search algorithm to find the extrema of a function.

Before describing the newly proposed technique to adapt the mesh to improve the solution with regards to the path constraint, it is first necessary to define the general optimal control problem and the specific

collocation method being used herein.

A. Optimal Control Problem Definition

In direct collocation methods, the optimal control problem is posed in terms of τ, which is related to time, t, by

$$\tau = \frac{2t - (t_f + t_0)}{t_f - t_0}. \tag{1}$$

In the Bolza form of the optimal control problem, the cost functional is defined with a Mayer part, Φ, and a Lagrangian part, $\mathscr{L}$, in continuous time as

$$J = \Phi(\mathbf{x}(-1), t_0, \mathbf{x}(1), t_f) + \frac{t_f - t_0}{2} \int_{-1}^{1} \mathscr{L}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) \tag{2}$$

where above $\mathbf{x}$ is a vector of state variables. This is subject to the dynamic constraints

$$\frac{d\mathbf{x}}{d\tau} = \frac{t_f - t_0}{2} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau), \tag{3}$$

the boundary conditions

$$\mathbf{B}(\mathbf{x}(-1), t_0, \mathbf{x}(1), t_f) = \mathbf{0}, \tag{4}$$

and the path inequality constraints

$$\mathbf{C}(\mathbf{x}(\tau), \mathbf{u}(\tau)) \leq \mathbf{0}. \tag{5}$$

This form of the optimal control problem may be transcribed to an NLP using one of several popular methods.

III. Direct Collocation Method

A. Legendre-Gauss-Radau Collocation

While there exist many varieties of direct collocation methods, the Legendre-Gauss-Radau (LGR) collocation scheme[14] is implemented in this study. Although a thorough description of the method is provided in the reference, only a short discussion is provided here. In the LGR method, collocation points lie on the interval $\tau \in$ [-1,1) where the n points are chosen as the roots of the Legendre polynomials, $P_{n-1}(\tau) + P_n(\tau)$. An additional, "noncollocated" $\tau_{n+1} = 1$ point is included in the approximation. A state variable, say x, may be approximated on this interval using Lagrange polynomial interpolation via the relation

$$x(\tau) \approx \sum_{i=1}^{n+1} x_i L_i(\tau) \tag{6}$$

where the Lagrange polynomial basis is

$$L_i(\tau) = \prod_{\substack{j=1 \\ j \neq i}}^{n+1} \frac{\tau - \tau_j}{\tau_i - \tau_j}, \quad i = 1, ..., n+1. \tag{7}$$

It is then possible to approximate the derivative of the state at each discretized τ_k using the differentiation matrix, D, which is equivalent to the rate of change of the Lagrange polynomial.

$$\dot{x}(\tau_k) \approx \sum_{i=1}^{n+1} x_i \dot{L}_i(\tau_k) = \sum_{i=1}^{n+1} D_{ki} x_i \tag{8}$$

The differentiation matrix is distinct for each collocation scheme being used. For the LGR method it is an (n x $n+1$) matrix. The (n x n) submatrix is inversely related to the integration matrix as described in the reference. Using the LGR differentiation matrix to calculate the derivative of a state or control on a set of points within an interval with spacing defined by the LGR basis is considerably more accurate than fixed

spacing methods (such as the 3 or 5 point formulas) with the same number of collocation points because it uses information from the entire interval, i.e., it is a global method over the current interval.[15] The differentiation matrix is a linear operator, and the discretized state derivative may be expressed in matrix multiplication form as

$$\dot{\mathbf{x}} \approx \mathbf{D}\mathbf{x}. \tag{9}$$

The control, u, may be similarly approximated. Referring to Equation 3 the dynamics of the problem may then be constrained by the relation

$$\mathbf{D}\mathbf{x} - \mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{0}, \tag{10}$$

where $\mathbf{x}$ and $\mathbf{u}$ represent the discretized state and control vectors at every collocation point. The constraint is enforced at each collocation point, excluding the noncollocated point τ_{n+1}. Therefore there are n equality constraints associated with the dynamics for each state. The path constraints from Equation 5 are also discretized and constrained at all collocation points, adding another $n+1$ constraints to the NLP. These may be either equality or inequality constraints, depending on the problem.

B. Mesh Intervals

As described in the Introduction, in order to improve accuracy and convergence time of a solution, the collocation point mesh may be modified to better capture regions of interest. In order to increase accuracy one may use a p-method by inserting additional points, requiring a new spacing for the LGR grid. The h-method method involves splitting the time interval into multiple intervals, each within the LGR interval of $\tau \in$ [-1,1). These intervals are then linked by setting the last point, which is considered noncollocated in the LGR grid, to be the first point in the following segment. This is particularly useful for problems where a sudden change may occur which modifies the dynamics, such as a staging rocket shedding mass and changing thrust. Adding intervals may serve to increase the sparsity of the NLP, as is pointed out by Darby et al.[12,13] Additional constraints must be added to the NLP to link the state and time variables between intervals. As mentioned previously, many schemes seek to adapt the mesh intervals and number of points to reduce error in the state. However, the following algorithm is proposed to detect violations in the path constraints between points and adapt the mesh accordingly.

IV. Mesh Adaptation

Typically a path constraint is expressed as an inequality, e.g. the heat rate must remain below a certain value, or the aircraft must remain outside a certain geographic zone. This means that the actual value of the path will vary within the trajectory, but will not be allowed to exceed the upper or lower limits of the constraint at the collocation points. However, because the optimal solution often borders these limits, the NLP solver may sometimes attempt to push the solution to exactly meet the path constraint at certain points. Unfortunately, this likely means that between these points the solution violates the constraint, although this is unknown to the NLP solver. It is therefore beneficial to perform post-solution mesh adaptation in order to ensure that the path constraints are met within a certain degree of tolerance. The current method performs first h then p adaptation, depending on the type of error present in the path constraint.

A. *h*-Method

The algorithm begins by applying the differentiation matrix operator to the discretized path constraint as

$$\dot{\mathbf{C}}(\mathbf{x}(\tau), \mathbf{u}(\tau)) = \mathbf{D}\mathbf{C}(\mathbf{x}(\tau), \mathbf{u}(\tau)). \tag{11}$$

This gives an indication of how quickly the path constraint is changing. This derivative is an approximation represented by the Lagrange interpolating polynomial

$$P_{\dot{C}}(\tau) \approx \sum_{i=1}^{n+1} \dot{\mathbf{C}}(\mathbf{x}(\tau), \mathbf{u}(\tau)) L_i(\tau). \tag{12}$$

It is possible to identify points in the segment where the time derivative of the path is zero by solving for the roots of the interpolating polynomial, or where $P_{\dot{C}}(\tau) = 0$. This may be done numerically by finding

the eigenvalues of the corresponding companion matrix.[16] These points, τ_0, after complex values or values outside the interval from [-1,1] are discarded, are candidate extrema. The value of the path is calculated at each valid root and checked to see if it violates the path constraints. If so, the algorithm divides the current segment at the violating root. The sum of the number of points in the two new intervals is set as the original number of points subject to a minimum of four points per segment. This avoids unnecessary growth of the order of collocation points and mirrors the approach for mesh division from Patterson et al.[11]

B. *p*-Method

In the p method, a different criterion for error is proposed. The roots of the polynomial of the derivative of the path constraint, τ_0, represent extremal points; they indicate that the path constraint might be at a maximum, a minimum, or an inflection point. The h-method evaluates these points against the user supplied upper and lower bounds to check if the constraint is violated. In the p-method, the same points are checked for error in the interpolation, which is directly related to the number and spacing of the points. There are two ways to evaluate the value of the path constraint at the extrema. First, the path constraint may be calculated from the state and control values at all the current points in the interval, after which the path constraint can then be interpolated at the extrema points. This is written as

$$\hat{\mathbf{C}}(\tau_0) \approx \sum_{i=1}^{n+1} \mathbf{C}(\mathbf{x}(\tau_i), \mathbf{u}(\tau_i)) L_i(\tau_0). \tag{13}$$

Alternatively, each of the state and control values may be first interpolated to the extrema points, and the path constraint calculated at those points. This approximation of the path constraint at the extrema points is

$$\bar{\mathbf{C}}(\tau_0) \approx \mathbf{C}\left(\sum_{i=1}^{n+1} \mathbf{x}(\tau_i) L_i(\tau_0), \sum_{i=1}^{n+1} \mathbf{u}(\tau_i) L_i(\tau_0)\right). \tag{14}$$

The difference between the two methods lies in the order of interpolation; whether the interpolation is performed on the states and controls, or whether it is performed on the path constraint itself. The interpolation approximation of the path constraint function (Equation [13]) is more error prone than the function evaluated at the interpolation approximation of its arguments (Equation [14]). However, as the interpolation improves by adding more points into the mesh, the error in the approximation is decreased. In this sense, the path constraint serves as a test function upon which we can judge the goodness of the mesh; in fact, it is an ideal test function because it is itself a parameter of interest.

The error in the path constraint is then calculated as

$$e_P = \left\| \frac{|\bar{\mathbf{C}}(\tau_0) - \hat{\mathbf{C}}(\tau_0)|}{1 + |\bar{\mathbf{C}}(\tau_0)|} \right\|_\infty. \tag{15}$$

This error is calculated for all extrema and all path constraints, and the maximum is taken to be a surrogate for the overall error in the interval. The error is checked against a user defined threshold, ϵ_P. If the threshold is exceeded, the number of collocation points in the interval, N, is increased by Q using the method described by Patterson et al,[11]

$$Q = log_N\left(\frac{e_P}{\epsilon_P}\right) \tag{16}$$

If it is determined that the interval is to be divided via the h-method, the p-method is not applied because a new point will already be inserted directly at the extremal point. However, if no constraint violations were detected but the path constraint error e_P exceeds the threshold, the new points will be added to the interval. Once all intervals have been checked for both h and p updates, the NLP is rerun using the previous solution as a new guess. The new solution is rechecked, and if the path constraint violations fall within the user tolerance the operation is terminated. One particular benefit of this method is that it automatically identifies the most logical points where the solution should be checked for path constraint violations and interpolation error, thereby minimizing unnecessary calculations on points which are not extrema.

This method may be used in conjunction with any other method which reduces error on the state or control vectors. It should be performed only once a converged NLP solution is obtained on a fully converged mesh, since the best approximation to the path constraint will only be available for an accurate resolution of the state and controls. The flow diagram in Figure 1 is provided as a visual description of the algorithm.

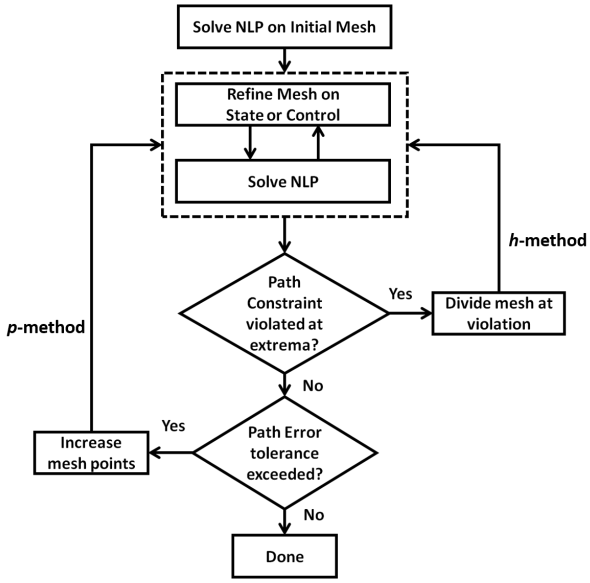

Figure 1. Flow Diagram illustrating the Path Constraint Adaptive Mesh Algorithm.

V. Examples

A. Keep-Out Zones Example Problem

A common problem in optimal control is to minimize the time required for a vehicle to travel between two points while avoiding keep-out zones. This example might represent a Mars rover avoiding terrain obstacles, a jet aircraft avoiding radar missile sites, or a hypersonic vehicle avoiding population dense areas. The objective is

$$minimize\ J = t_f, \tag{17}$$

where the final time, t_f, is determined by the final boundary condition when the vehicle reaches the target location (x_f, y_f),

$$\mathbf{B}(t_f) = \begin{bmatrix} x(t_f) - x_f \\ y(t_f) - y_f \end{bmatrix} = \mathbf{0}. \tag{18}$$

The vehicle must obey a set of dynamics described by differential equations,

$$\begin{aligned} \frac{dx}{dt} &= V\cos\psi \\ \frac{dy}{dt} &= V\sin\psi \\ \frac{d\psi}{dt} &= u. \end{aligned} \tag{19}$$

Above, V is a constant velocity, x and y are the vehicle coordinates, ψ is the vehicle's heading, and u is the control to be obtained by the solution of the optimal control problem. The vehicle must avoid two keep-out zones; for this example represented simply as two circular regions with a 1 km radius centered at kilometers (3,1) and (5,3) in the x and y coordinates. The keep-out zones are implemented as path constraints using the Euclidean distance. The constraint corresponding to the kth keep-out zone is

$$\mathbf{C}(\tau_i) = r_{ko,k} - ((\mathbf{x}(\tau_i) - x_{ko,k})^2 + (\mathbf{y}(\tau_i) - y_{ko,k})^2)^{\frac{1}{2}} \leq 0. \tag{20}$$

Note that $\mathbf{C}(\tau_i)$ is a vector corresponding to each collocation point τ_i. The variables $x_{ko,k}$, $y_{ko,k}$, and $r_{ko,k}$ are the coordinates and radius of the keep-out zones.

1. *No Path Mesh Adaptation*

The problem has been formed in MATLAB using the LGR collocation method using 5 collocation points (plus the noncollocated point at +1) to transcribe to an NLP, which was then solved using SNOPT with no subsequent mesh refinement. The trajectory is shown in Figure 2. Lacking a check on the path constraint, the solution passes right through the two keep-out zones because the collocation points happen to fall outside the constraint radius. Blindly adding more points at the expense of increased computation time tends to alleviate the problem, as shown in Figure 3, although again the solver has found a solution which violates the path constraints between collocation points. Clearly, the solver has in fact converged to a solution which cuts through the keep out zones in order to minimize the detour the vehicle must take around them.

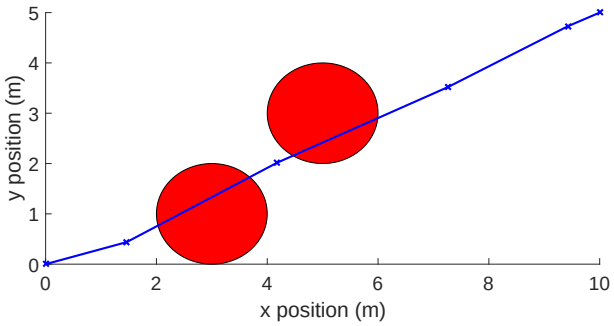

Figure 2. Solution found using an LGR mesh with 6 total points. Because the path constraints fall in between the points, the NLP solver reports that an optimal solution has been achieved.

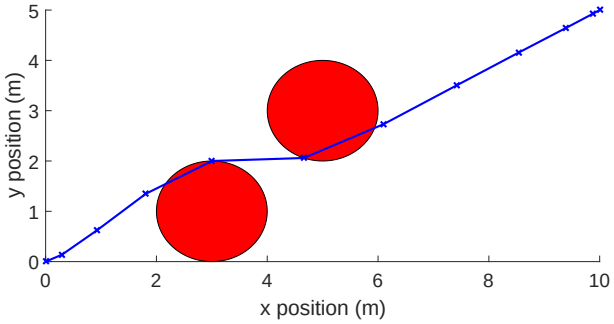

Figure 3. Solution found using 12 total points. Again, the NLP solver reported an optimal solution, despite the obvious path constraint violation between collocation points.

2. *Path Mesh Adaptation*

The values of the two path constraints at the collocation points after the first NLP solution are shown in Figure 5, along with the $\hat{C}$ Lagrange interpolation approximation between the points. It is seen that the collocated points do not fall under the lower bound of 1 km, thus the NLP reports a converged solution. Interestingly, the $\hat{C}$ interpolated values don't fall below the lower bound either. The derivative of the path was calculated via Equation 9 and has been plotted in Figure 4. Each of the interpolated polynomials has only one root, indicating that the trajectory has a single minimum approach distance from each keep-out zone. The interpolation predicts these to be at τ of -0.438 and -0.003, or translated to time at t of 5.26 seconds and 9.33 seconds. The algorithm then evaluates the value of $\bar{C}$, given by Equation 14, at those points. They are both found to be in violation of the lower bound as shown in red in Figure 5. Therefore, the mesh is split into three intervals of four collocated points each, and the problem is re-transcribed and sent back to the NLP solver.

The resulting solution following the first mesh iteration is shown in Figure 6. The algorithm is again applied, finding that there remains a path constraint violation in the third segment. The segment is divided, transcribed, and resolved resulting in the solution shown in Figure 7. The path constraint violation is found

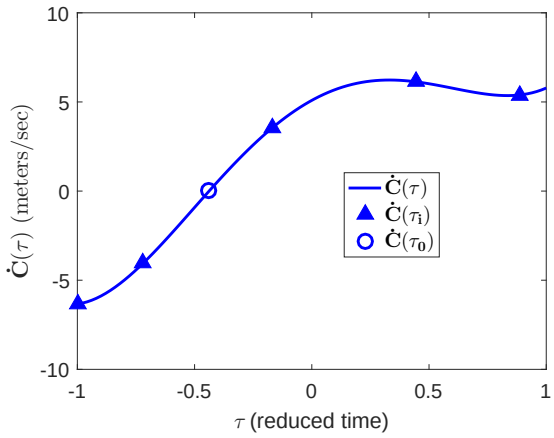

Figure 4. The rate of approach to the keep-out zones is simply the time derivative of the path constraints. Where this derivative is zero, there is the possibility of extrema.

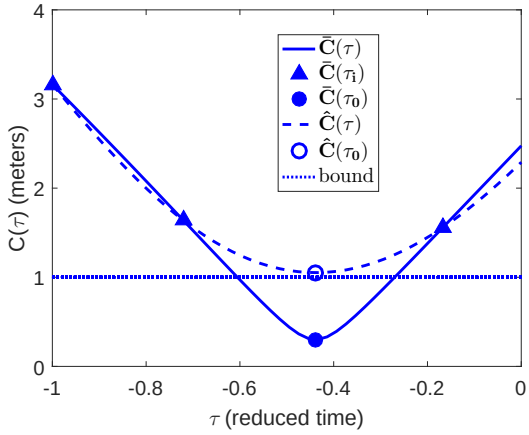

Figure 5. The path constraints at the collocation points and the $\hat{C}$ Lagrange interpolation do not show a violation of the lower limit. The $\bar{C}$ value of the minimum has been plotted in red.

here to be less than the user supplied tolerance of 5%, thus the iterations are ceased. Note that no iterations have been performed to reduce error in the state or control, but this could easily be done at the same time as the path constraint adaptation. They have been omitted here to clarify the process.

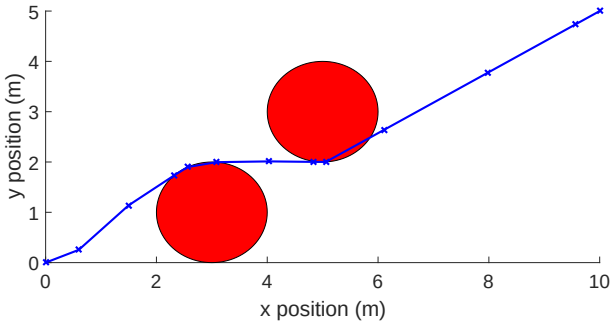

Figure 6. The first mesh adaptation identified two path constraint violations, thus dividing the mesh into three intervals of four points each.

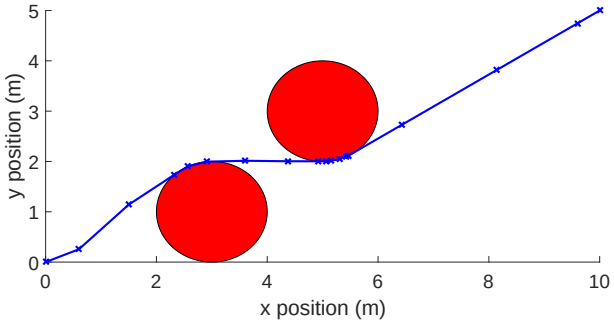

Figure 7. The second mesh adaptation identified one path violation, therefore one additional interval was inserted raising the total to four intervals of four points each.

In this example the p-method was not required because constraint violations corresponding to refinements using the h-method are checked first. This is not the case generally.

B. Space Shuttle Reentry Example Problem

Another interesting example problem is presented by Betts,[4] in which a reusable reentry vehicle (such as the Space Shuttle) attempts to maximize its crossrange while not exceeding a safety limit imposed for heat generated on the leading edge of the wing. This problem presents another opportunity to demonstrate the usefulness of the path constraint mesh adaptation.

The optimal control problem is written as

$$minimize\ J = -\theta(t_f) \tag{21}$$

Subject to the dynamic constraints

$$\begin{aligned}
\frac{dh}{dt} &= V\sin\gamma \\
\frac{d\theta}{dt} &= \frac{V}{r}\cos\gamma\cos\psi \\
\frac{d\phi}{dt} &= \frac{V\cos\gamma\sin\psi}{r\cos\theta} \\
\frac{dV}{dt} &= -\frac{D}{m} - g\sin\gamma \\
\frac{d\gamma}{dt} &= \frac{L\cos\beta}{mV} + \cos\gamma\left(\frac{V}{r} - \frac{g}{V}\right) \\
\frac{d\psi}{dt} &= \frac{L\sin\beta}{mV\cos\gamma} + \frac{V}{r}\cos\gamma\sin\psi\tan\theta
\end{aligned} \tag{22}$$

where above h is altitude (m), θ is latitude (rad), ϕ is longitude (rad), V is velocity (m/s), γ is the flight path angle (rad), ψ is the heading (rad), m is the mass (kg), g is the gravitational acceleration, and L and D are lift and drag, respectively. Parameters to calculate lift, drag, and aerodynamic heating, q, are provided in the reference. The geometric radius, r, is the altitude plus the radius of the earth. The controls being sought are the vehicle's angle of attack, α, and its bank angle, β.

A path constraint is added to ensure the aerodynamic heating, q, does not exceed 70 BTU/ft^2/sec,

$$C(\tau) = q - 70 \leq 0. \tag{23}$$

The vehicle begins its trajectory on the prime meridian at 260 kft traveling east along the equator at 25.6 kft/sec, with a flight path angle of -1 degree. The final point occurs at the terminal area energy management (TAEM) location set as an altitude of 80 kft and velocity of 2.5 kft/sec.

The optimal control problem presented above was solved using GPOPSII,[5] a generalized direct collocation software tool implemented in MATLAB. The software allows the user to select several options for scaling, derivative calculations, NLP solvers, and mesh adaptation algorithms. The algorithm by Patterson et al[11] has been used to adapt the mesh for the states and controls to a tolerance of 0.001. The first solution, before applying path constraint mesh adaptation, is displayed in Figures 8 to 13. This solution matches well with the trajectory and controls presented by Betts,[4] where the vehicle begins by sharply banking to nearly 80 degrees while pulling up to quickly turn the heading in order to maximize the cross-range distance, achieving over 30 degrees of latitude before arriving at the TAEM interface.

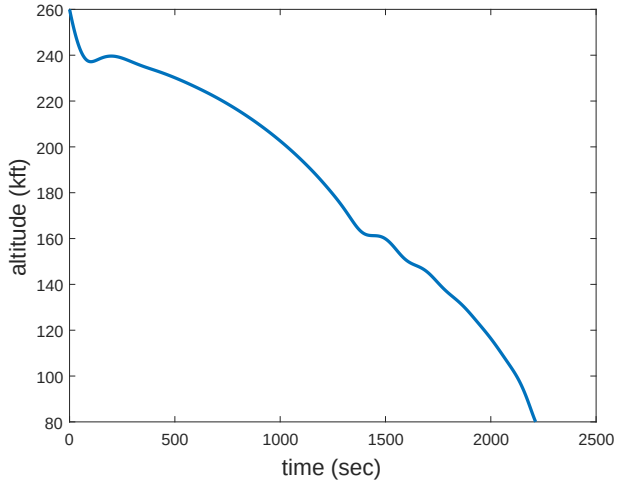

Figure 8. Altitude (m)

The heating profile prior to path constraint mesh adaptation, shown in Figure 14, appears to obey the maximum constraint of 70 BTU/ft^2/sec. However, on closer inspection it can be seen that the interpolated solution has an overshoot of the limit which was not captured by the state-based mesh adaptation algorithm,

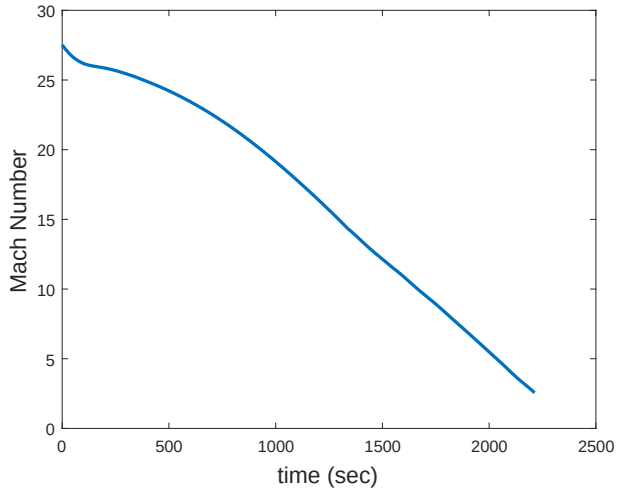

Figure 9. Mach number

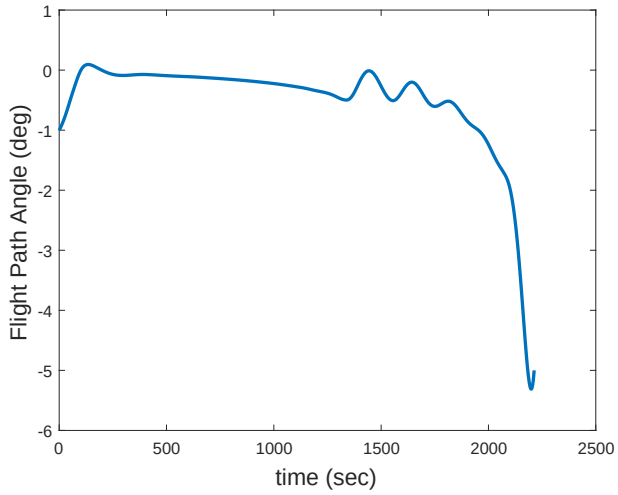

Figure 10. Flight path angle (deg)

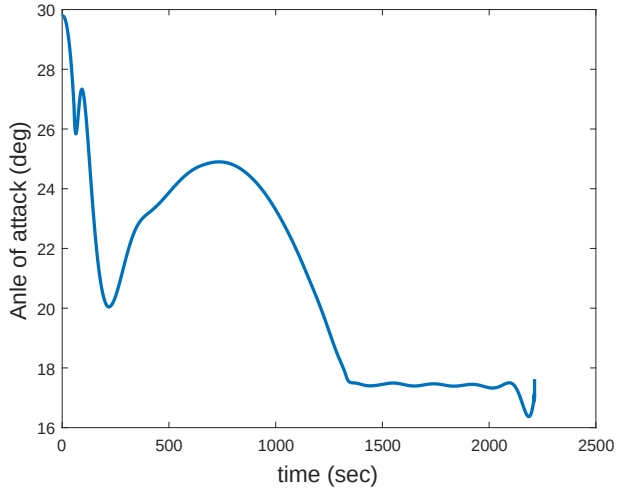

Figure 11. Angle of attack (deg)

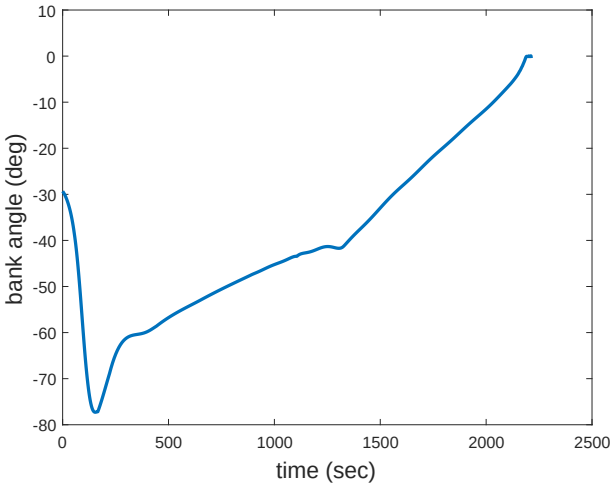

Figure 12. Bank angle (deg)

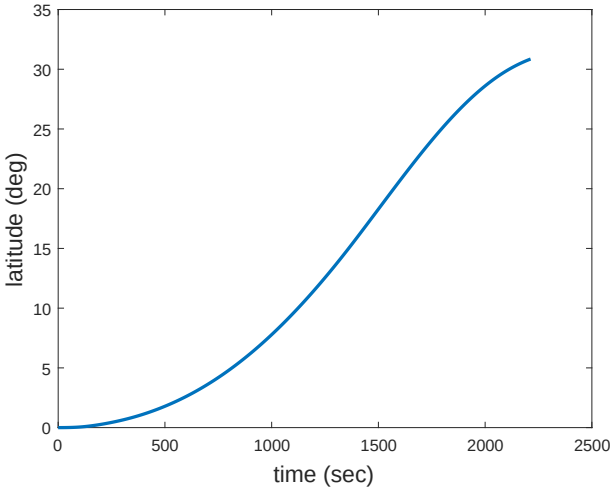

Figure 13. Latitude (deg) representing the amount of crossrange maximized in the optimal control problem solution.

despite already being divided into 13 intervals. The first pass of the path constraint mesh adaptation algorithm divided the mesh interval at the maximum point shown in Figure 15 (h-method), and also added additional points to several intervals in order to decrease the error where the heating value traced the upper limit (p-method).

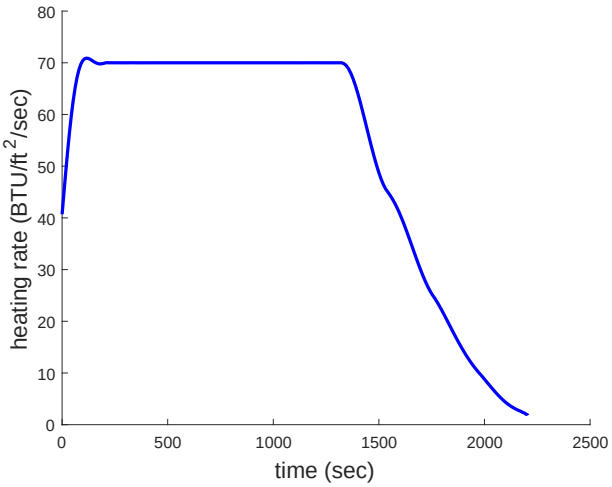

Figure 14. Heat rate along the leading edge of the wing. A close inspection reveals a small overshoot as the heat reaches the maximum allowable value of 70 BTU/ft^2/sec

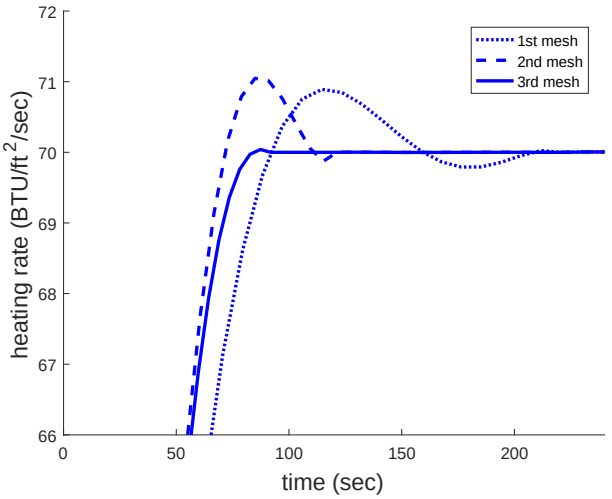

Figure 15. A closeup view of the heat rate overshoot illustrates the improvements to the solution with meshing.

After the first path constraint mesh adaptation, the NLP solver was run a second time. Once again the path constraint was checked, and once again the interval was divided where the peak heating overshot the limit. Additional points are also added after checking the interpolation error. After one final solution of the NLP, application of the algorithm results in no errors as shown in the solid line solution seen in Figure 15.

VI. Conclusion

The example problems illustrate that sometimes state or control based mesh adaptation schemes do not correctly regulate the behavior of a path constraint between collocation points. Experience has suggested that this can be a problem, particularly in control problems where the optimal solution trajectory rides along the path boundaries. In order to efficiently find the best point to modify the mesh, the derivative of the path is approximated using the differentiation matrix, and extrema values are found and checked for constraint violations. To remedy these violations the mesh is split at these points (h-method). A second type of error is

defined based on the comparison of two interpolation schemes. In this case the mesh is refined by increasing the number of points used in the polynomial approximation (p-method). Because the scheme proposed in this work gives priority to dividing the mesh, the algorithm may be classified as an hp-method, as opposed to a ph-method. It may be possible to extend this method to checking the upper and lower limits of state and controls, in order to ensure they do not in a similar way violate the bounds at the extrema between points. Another application may be to use the extrema points to check tolerances on the equality constraints associated with the equations of motion using some test function, possibly a power of the state or control.

References

[1]Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Review*, Vol. 47, No. 1, 2005, pp. 99–131. doi: 10.1137/S0036144504446096

[2]"MATLAB and Optimization Toolbox Release 2014a, The MathWorks, Inc., Natick, Massachusetts, United States".

[3]Boyd, J., *Chebyshev and Fourier Spectral Methods: Second Revised Edition*, Dover Books on Mathematics, Dover Publications, 2001, Chapter 4.

[4]Betts, J., *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, Society for Industrial and Applied Mathematics, 2nd ed., 2010, Chapters 4 and 6.

[5]Patterson, M. A. and Rao, A. V., "GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming," *Association for Computing Machinery (ACM) Transactions on Mathematical Software*, Vol. 41, No. 1, Oct. 2014, pp. 1:1–1:37. doi: 10.1145/2558904

[6]Liu, F., Hager, W. W., and Rao, A. V., "Adaptive mesh refinement method for optimal control using nonsmoothness detection and mesh size reduction," *Journal of the Franklin Institute*, Vol. 352, No. 10, 2015, pp. 4081 – 4106. doi: http://dx.doi.org/10.1016/j.jfranklin.2015.05.028

[7]Betts, J., *Sparse Optimization Suite (SOS), User's Guide*, Applied Mathematical Analysis, LLC, www.appliedmathematicalanalysis.com/resources/software, 2013, Retrieved from the internet 29 January 2015.

[8]Becerra, V., "Solving complex optimal control problems at no cost with PSOPT," *Computer-Aided Control System Design (CACSD), 2010 IEEE International Symposium on*, Sept 2010, pp. 1391–1396. doi: 10.1109/CACSD.2010.5612676

[9]Fahroo, F. and Ross, I. M., "Advances in Pseudospectral Methods for Optimal Control," Guidance, Navigation, and Control and Co-located Conferences, American Institute of Aeronautics and Astronautics, Aug 2008. doi: http://dx.doi.org/10.2514/6.2008-7309

[10]Gong, Q., Fahroo, F., and Ross, M., "Spectral Algorithm for Pseudospectral Methods in Optimal Control," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, 208, pp. 460–471. doi: http://dx.doi.org/10.2514/1.32908

[11]Patterson, M. A., Hager, W. W., and Rao, A. V., "A ph mesh refinement method for optimal control," *Optimal Control Applications and Methods*, Vol. 36, No. 4, 2015, pp. 398–421. doi: 10.1002/oca.2114

[12]Darby, C. L., Hager, W. W., and Rao, A. V., "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications and Methods*, Vol. 32, No. 4, 2011, pp. 476–502. doi: 10.1002/oca.957

[13]Darby, C. L., Hager, W. W., and Rao, A. V., "Direct Trajectory Optimization Using a Variable Low-Order Adaptive Pseudospectral Method," *Journal of Spacecraft and Rockets*, Vol. 48, No. 3, May 2011, pp. 433–445. doi: http://dx.doi.org/10.2514/1.52136

[14]Garg, D., Patterson, M. A., Francolin, C., Darby, C. L., Huntington, G. T., Hager, W. W., and Rao, A. V., "Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a Radau pseudospectral method," *Computational Optimization and Applications*, Vol. 49, No. 2, 2011, pp. 335–358. doi: 10.1007/s10589-009-9291-0

[15]Trefethen, L., *Spectral Methods in MATLAB*, Society for Industrial and Applied Mathematics, 2000, Chapter 1.

[16]Trefethen, L. and Bau, D., *Numerical Linear Algebra*, Society for Industrial and Applied Mathematics, 1997, Chapter 25, pp 191-192.

IX. Appendix B

The following document was prepared for publication at the 2017 American Controls Conference. It was not deemed of sufficient interest for publication, so will be archived here.

Direct Orthogonal Collocation Optimal Control Using Estimation-Based Mesh Adaptation

Ryan Carr and Richard Cobb
Department of Aeronautics and Astronautics
Air Force Institute of Technology
Wright-Patterson Air Force Base, Ohio 45433
Email: ryan.carr@afit.edu

Meir Pachter and Scott Pierce
Department of Electrical and Computer Engineering
Air Force Institute of Technology

***Abstract*—Mesh adaptation is a critical part of optimal trajectory generation via the Direct Orthogonal Collocation method of optimal control. While a converged nonlinear programming problem indicates that error thresholds are satisfied at the collocation points, adaptive meshing schemes typically focus on reducing discretization error, or equivalently, interpolation error between the points. The choice of error parameters which signal mesh refinement is not trivial and if chosen poorly can adversely affect the quality of the optimal solution. Therefore it is necessary to choose parameters which are in tune with the actual application of the calculated optimal control trajectory. One use for an optimal control solution is as an open-loop reference trajectory for a traditional state feedback controller which requires state estimation. In this case it may be beneficial to ensure that the optimal trajectory is sufficiently resolved to be tracked by the high frequency state estimator, meaning that errors in the interpolation of the optimal control do not propagate forward into large errors in the state estimate. In the current study, a new error parameter is employed based on the Mahalanobis Distance, which is calculated from the estimated error covariance matrix provided by the estimator, and is used to adapt the collocation mesh of the optimal control problem in a natural and convenient way to ensure that interpolation of the open-loop trajectory does not create unacceptable errors in the propagation step of the estimator. An example problem is provided to demonstrate the application of this technique.**

I. Introduction

In the collocation method for direct optimal control, the control and state trajectory are obtained by transcribing the continuous objective function, dynamic constraints, states and controls to a set of discrete collocation points and solving the resulting nonlinear programming problem with well established techniques such as Sequential Quadratic Programming (SQP) [1]. This is a good alternative to indirect methods which require solution of a Two-Point Boundary Value Problem (TPBVP) which may be infeasible for many realistic problems, particularly in the presence of state constraints. If the spacing of the collocation points are distributed using a basis of orthogonal Legendre polynomials, the collocation method can exhibit near spectral accuracy [2]. Often called the Pseudospectral Method [3], here the technique is termed direct orthogonal collocation as it is not necessarily limited to any specific set of orthogonal polynomials.

In direct collocation, the dynamics are enforced by equality constraints which must be satisfied within a desired tolerance at the collocation points. Lagrange interpolation is used to determine the value of the states and controls between points, and is thus subject to error due to "overmodeling" depending on the number and spacing of the points. In some instances, large discretization errors may be present between points for a fully converged NLP solution because the quality of the solution is exclusively assessed at the node points. This necessitates refinement of the number and location of the points in the mesh [4].

In direct orthogonal collocation there are two possibilities for adapting the point placement. First, the Cauchy Interpolation error theorem predicts that increasing the number of total points used in the interpolating polynomial can serve to reduce the error, although this may introduce undesirable oscillations near the endpoints due to the Runge phenomenon [5]. The second method divides the solution space into distinct intervals, linking together multiple polynomials. Additional constraints are then added to the NLP to link the intervals together. These two mesh refinement techniques are commonly called the *p*-method and the *h*-method, respectively.

A number of schemes for refining orthogonal collocation meshes using these two basic techniques have been proposed. One adaptation algorithm uses the differentiation matrix, D, to calculate the time derivative of the control. At a point where the control derivative is larger than some threshold value, a "knot" is placed to subdivide the mesh into two intervals [6]. Another author suggests a *ph*-method in which the error is defined by the difference between the interpolated state and an integrated approximation. The number of points is first augmented, and if the polynomial exceeds some limit, the interval is divided [7]. A refinement on this method monitors the second derivative of the state through successive refinements to estimate the smoothness of the intervals [8]. Intervals which are non-smooth and have high error are divided, while smooth intervals with low error may be recombined and the number of points reduced.

Another scheme calculates the error in the dynamics at the midpoint between two points using the differentiation matrix, then either splits the interval or adds a fixed number of points depending on whether the error detected exceeds a local or global threshold [9]. A further method uses a similar error, but evaluated at a large number L of uniformly spaced

points interpolated over the mesh interval (e.g. L=1000). If the error exceeds a threshold the curvature of the state or control is calculated using the first and second derivatives. If the curvature value is too high the interval is divided, otherwise the polynomial order is increased [10].

A common element of all these refinement algorithms is that they define the error by parameters related to the Legendre based point spacing. For example the differentiation matrix, which is defined by the number of points and the particular Legendre basis used for spacing, is inherently tied to the transcription process. The approximation of the derivative of the state or control obtained using this matrix reflects the errors in the discretized solution. Therefore, it makes sense to use a transcription-based parameter to signal that mesh refinement is necessary to ensure an adequate approximation of the continuous optimal control solution.

The purpose of the current work is not to detract from these methods, but instead to add another layer to them, cognizant of the ultimate purpose of an optimal control solution: It is to be used as an open-loop reference trajectory to control the movement of a vehicle over time [11]. When used in this fashion it may happen that over time the vehicle will stray from the reference trajectory due to several factors. First, it is unlikely that the dynamics used to calculate the optimal control solution match perfectly with reality. This is normally corrected using some type of feedback controller, with the possibility of real-time updates of the optimal control trajectory as more information becomes available [12]. An example controller using this type of scheme is shown in Figure 1. A major source of error is in the interpolation of the optimal control solution to a time-step which is suitable for the on-board system. Specifically, if the system uses a recursive state estimator such as a Kalman filter, during the propagation step the filter must interpolate the optimal control solution to obtain the required control at the current time step. The estimated state is then propagated forward in time using the state transition matrix. If the interpolation of the control introduces an error, this process will propagate errors into the state estimate which must then be corrected with feedback control, and a vicious cycle may arise. For this reason it is proposed that the mesh refinement of the optimal control solution be modified to include a new type of error that represents the mismatch between the direct collocation mesh and the uniformly spaced time-steps taken by the state estimator. Although the interaction with other control system components may also be important, this study focuses on the interaction between the optimal control solution and the propagation step of the estimation filter, whose modules have been highlighted the control system's block diagram in Figure 1.

II. Motivating Example

As a simple example of why it may be useful to adapt the mesh to improve the performance of the estimator, an optimal control problem is analyzed where a vehicle must traverse a

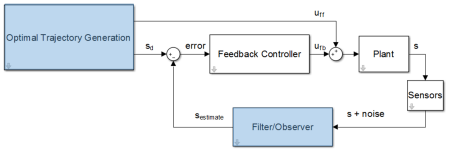

Fig. 1: The interaction of the optimal trajectory generation with the state estimator (filter/observer).

field with keep-out constraints to arrive at a target location. The objective is

$$\underset{u(t)}{minimize} \quad J = t_f \tag{1}$$

The motion of the vehicle, constrained to the horizontal plane, is described by the dynamics

$$\begin{aligned} \dot{x} &= Vcos(\theta) \\ \dot{y} &= Vsin(\theta) \\ \dot{\theta} &= u, \quad -\frac{\pi}{2} \leq u \leq \frac{\pi}{2} \end{aligned} \tag{2}$$

Above, the x and y state are the vehicle's position in the plane expressed in meters, the velocity, V, is a constant set to 1.0 m/s^2, the heading, θ, is the angle off the x axis in radians, and the steering control is u expressed in rad/sec. The vehicle has a fixed initial and final state shown in Table I. The final heading and time are unconstrained.

TABLE I: Initial and final states of the vehicle.

x_0 (m)	y_0 (m)	θ_0 (deg)	x_f (m)	y_f (m)
0	0	45	10	10

There are additionally two circular keep-out zones the vehicle must avoid as it traverses the field. The location and radius of these zones are given in Table II.

TABLE II: Location and radius of the two circular keep-out zones.

$x_{KO,1}$ (m)	$y_{KO,1}$ (m)	$r_{KO,1}$ (deg)
2	3	2
$x_{KO,2}$ (m)	$y_{KO,2}$ (m)	$r_{KO,2}$ (deg) (m)
7	5	2

The above problem has been solved using the commercially available direct orthogonal collocation software *GPOPS-II*[13]. The keep-out zones are implemented as path constraints along the entire trajectory. The initial mesh is a single interval with 6 points using Legendre-Gauss-Radau collocation[14]. A solution to the NLP is quickly obtained, but the mesh must

then be refined three times before achieving an error tolerance of 1e-3 using the method from Patterson et al[7].

It is then supposed that the control solution will be used open loop by the vehicle as it traverses the field toward the target. The vehicle uses an Unscented Kalman Filter (UKF)[15] to propagate the estimate of its position and heading at a rate of 100 Hz. The propagation of the sigma points is performed using a 4 step Runge-Kutta integration. The control signal from the optimal solution is interpolated onto the uniform steps used by the UKF via Lagrange Interpolation. This is the process which is likely to introduce errors into the propagation, as the interpolated control may oscillate between collocation points as seen in Figure 2.

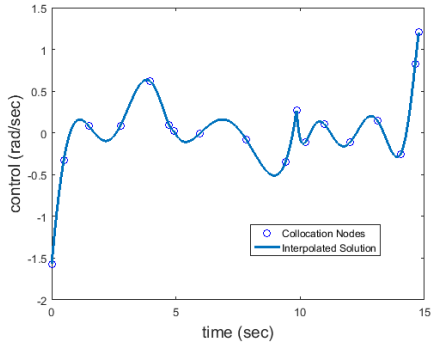

Fig. 2: The interpolated control signal to be used in open-loop propagation of the state estimator shows oscillation between collocation points, despite the fact that standard mesh adaptation has already been performed.

The process noise covariance matrix Q_k is calculated using a linearized form of the dynamics, and then added to the covariance propagation step[16]. For clarity, it is also assumed that no sensor measurements are available for the update step of the UKF. This means that errors in the state estimate induced by poor interpolation of the optimal control will propagate through time without correction.

The oscillations in the interpolated control signal seen in Figure 2 result in a poor propagation of the estimated state. The interpolated and propagated heading are shown in Figure 3, providing a visualization of the problem. The propagated heading oscillates such that the interpolated heading strays outside the displayed bounds, which correspond to one standard deviation. The x and y state estimates shown in Figure 4 are better due to the smoothing action of the double integration, although there is a noticeable difference between the interpolated and propagated states by the end of the trajectory. Figure 4 additionally shows covariance ellipses corresponding to one standard deviation estimation error in the propagated position. While the mean propagated position misses the final target state, perhaps this is not too far from what should be expected given the uncertainty codified by the large covariance ellipse bounds. For this reason, an algorithm has been developed which uses the estimated error covariance to indicate the need to refine the mesh.

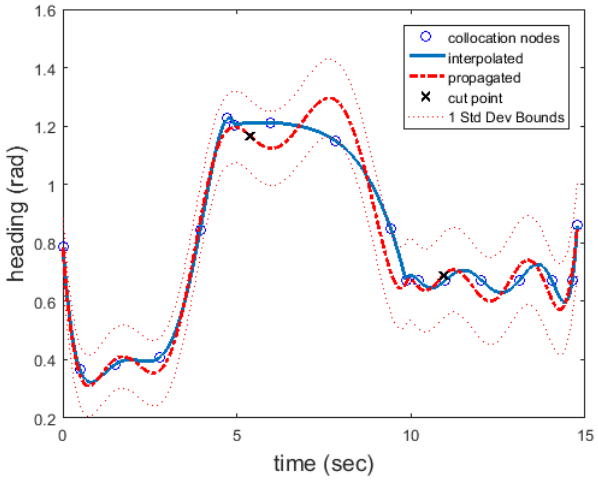

Fig. 3: The heading estimate diverges from the desired heading due to interpolation error.

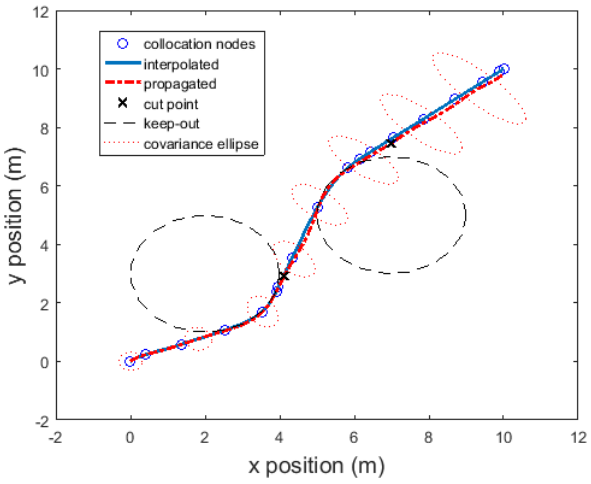

Fig. 4: The estimator predicted position of the vehicle gradually diverges from the optimal control desired solution due to interpolation error in the calculated optimal control produced by the numerical optimization algorithm. The covariance ellipse bounds represent one standard deviation.

III. The Algorithm

A. *Interpolation of the Optimal Control Trajectory*

Once the optimal trajectory is obtained comprising the state and control values at the collocation points, Lagrange interpolation is used to approximate the values of the state and control between points. When approximating the state x at some point τ, the Lagrange interpolation has the form

$$x(\tau) \approx \sum_{i=1}^{n+1} x_i L_i(\tau), \tag{3}$$

where the Lagrange polynomial basis is

$$L_i(\tau) = \prod_{\substack{j=1 \\ j \neq i}}^{n+1} \frac{\tau - \tau_j}{\tau_i - \tau_j}, \quad i = 1, ..., n+1. \tag{4}$$

The variable τ takes values between -1 and 1 over each mesh interval. It relates to the time, t, by the affine transformation

$$\tau = \frac{2t - (t_f + t_0)}{t_f - t_0}. \tag{5}$$

This interpolation operation is the source of the error which is addressed by the mesh refinement technique developed in this paper.

B. Definition of Error

Figures 3 and 4 display the mismatch between the states interpolated from the optimal control solution and states propagated through the UKF, but it is difficult to know whether or not this is acceptable. The state estimation error covariance matrix, $\mathbf{P}(t)$, which is propagated through the UKF, provides a natural measure of the amount of uncertainty expected in the propagated states. Therefore it is interesting to express the error in terms of the covariance by forming the Mahalanobis Distance (MD),

$$d_M(t_i) = \sqrt{(\mathbf{x}(t_i) - \hat{\mathbf{x}}(t_i))^T \mathbf{P}^{-1}(t_i)(\mathbf{x}(t_i) - \hat{\mathbf{x}}(t_i))}, \tag{6}$$

where above $\mathbf{x}(t_i)$ is the interpolated optimal control solution state vector at the i^{th} propagation timestep, and $\hat{\mathbf{x}}(t_i)$ is the estimator propagated state vector. This error is essentially a measure of how far the propagated state has strayed from the interpolation, but weighted by the fact that a certain amount of error is expected. The MD is a convenient, scalar measurement of error and for the example problem has been displayed in Figure 5.

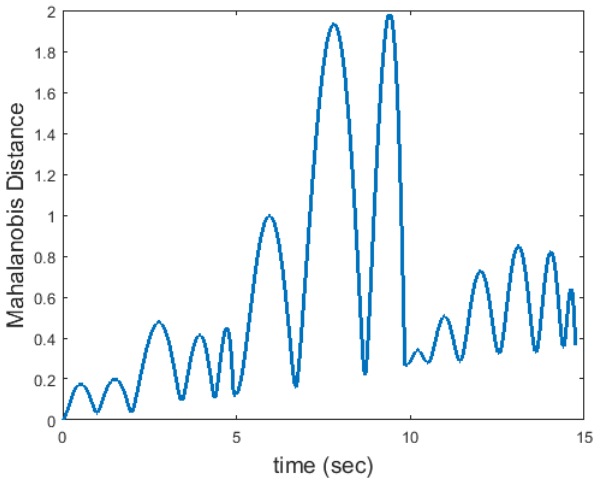

Fig. 5: The estimator-predicted heading diverges from the optimal heading due to interpolation error.

It is proposed that the MD be used as a mesh refinement error metric. To understand the significance of a specific value of the MD, it is helpful to assume that the error distribution about the propagated state is multivariate normal. In that case a surface of constant MD surrounding a point forms an ellipsoid centered at $\hat{\mathbf{x}}$. Then, the probability of any point belonging to this same distribution falling within the square of the MD follows a chi-square distribution with the number of states being, p, the degrees of freedom [17]. Therefore, points in $\mathbf{x}$ satisfying the relation

$$d_M^2 = (\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{P}^{-1} (\mathbf{x} - \hat{\mathbf{x}}) <= \chi_p^2(\alpha) \tag{7}$$

have a probability of 1 - α of being statistically equivalent to $\hat{\mathbf{x}}$. The assumption that the uncertainty of the propagated states matches a Gaussian distribution fits with the near-Gaussian nature of the UKF errors [15]. Therefore, to achieve a 99% probability that a point belongs to the propagated state the MD must be less than 0.34. These statistics are only meant to guide the user in proper selection of the limit of the value of MD, since the Gaussian assumption in the error is by no means assured.

C. Refining the Mesh

1) Locating the intervals where refinement is required: Once it has been determined that the MD threshold has been exceeded, refining the mesh can improve the interpolation of the optimal control. This is complicated by the particular structure of the collocation mesh, which is divided into intervals each containing a different number of collocation points. The general strategy is to reduce interpolation error by either adding more points or dividing the intervals.

A complicating factor is that the MD errors tend to originate at some time in the past and grow as time progresses. In other words, although the MD threshold was exceeded in interval I_i, the problem with the mesh actually originated in interval I_{i-1}, or I_{i-2}, etc. Therefore, rather than only refining the mesh in the current interval, it may be necessary to refine the mesh in one or more previous intervals. This can be done by searching backward in time for signs that the MD began to grow. The current algorithm allows the user to input a ratio of the threshold, such as one half, and the code seeks the earliest time the ratio of the threshold is reached. These two points, labeled t_d and $t_{d/r}$, may occur in the same interval, or in widely separated intervals. In the former case, the single interval containing both points is chosen for refinement. In the latter case, all intervals in between the two points are designated for refinement. Each interval is only considered once to avoid over-refining, regardless of the number of times the MD threshold is exceeded within the interval.

2) Adding Points: If the $t_{d/r}$ and t_d points occur in the intervals $I(t_{d/r})$ and $I(t_d)$, then it is likely that all intervals from $I(t_{d/r})$ to $I(t_d)$ require refinement. This is accomplished by adding an equal number of points to each interval, where the total number of points to be spread through the intervals is given by the relation

$$N = \left\lfloor \frac{I(t_d) - I(t_{d/r})}{\frac{t_d - t_{d/r}}{t_f - t_0}} \right\rfloor . \quad (8)$$

The number of points, N, is increased if the number of intervals between $I(t_d)$ and $I(t_{d/r})$ is large, and also if the relative time difference is small. This number is then spread evenly throughout the intervals marked for refinement. Each interval is then checked to ensure that the maximum number of points has not been exceeded. If so, the interval is split with the two new intervals containing half the maximum number of points.

3) Splitting Intervals: If t_d and $t_{d/r}$ occur in the same interval, then MD is increasing rapidly and thus the interval $I(t_d)$ is divided into two, with the number of points in each new interval being half the maximum number allowed.

In Figures 3 - 5, the points where the DM exceeds the threshold of 0.5 are displayed as small x's. Although inspection of Figure 5 reveals that the MD threshold is exceeded many times, the algorithm only identifies the first time the threshold is exceeded in an interval. Therefore only two cut points are indicated, corresponding to intervals 2 and 3, which are marked for refinement. Interval 1 is later marked for refinement because it contains the location where the MD exceeds the half-threshold value prior to exceeding the actual threshold in interval 2. In summary,

4) Algorithm Summary:

1) Solve the NLP on a coarse mesh.
2) Refine the mesh based on existing techniques.
3) Interpolate the state and control values onto a uniformly spaced mesh corresponding to the propagation steps of the estimator.
4) Propagate the state using the interpolated open-loop control.
5) Calculate the MD at each of the propagation time steps, compare to the threshold value set by the user.
6) Identify points, t_d, where the MD exceeds the threshold.
7) For each point, search backward in time to the point $t_{d/r}$ where the MD first reaches the user-supplied ratio of the threshold value.
8) Refine intervals by either adding points or dividing the interval.
9) Resolve the NLP on the newly refined mesh.

IV. Demonstration of the Algorithm

To demonstrate the algorithm, the refinement is performed on the mesh from the motivating example optimal control problem described by Equations (1) and (2) and Tables I and II. The Patterson mesh refinement method produced a mesh made of three intervals with 5, 4, and 6 collocation points respectively. The first mesh refinement using the current algorithm operates directly on this mesh. The control signal is interpolated using Equations 3 and 4, and the states are propagated using the UKF. When the Mahalanobis Distance error is calculated, it is found that all three intervals require refinement. In the resulting refinement step, two points are added to interval 1, while intervals 2 and 3 are each divided at the midpoint. The NLP is then reformulated and solved on this new mesh.

The subsequent mesh refinements then proceed four more times, as displayed in Figure 6. The MD error calculated after solving the NLP on the final mesh no longer exceeds the threshold limit of 0.5 as seen in Figure 7, and thus the mesh refinement is terminated. There remain oscillations in the MD beginning at approximately 6 seconds, but they do not exceed the threshold value.

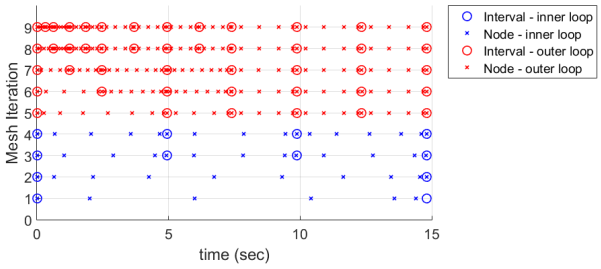

Fig. 6: After the initial refinement phase, the mesh is refined 5 more times in order to bring the Mahalanobis Distance below the threshold of 0.5. The final mesh consists of 12 intervals, each with a varying number of points.

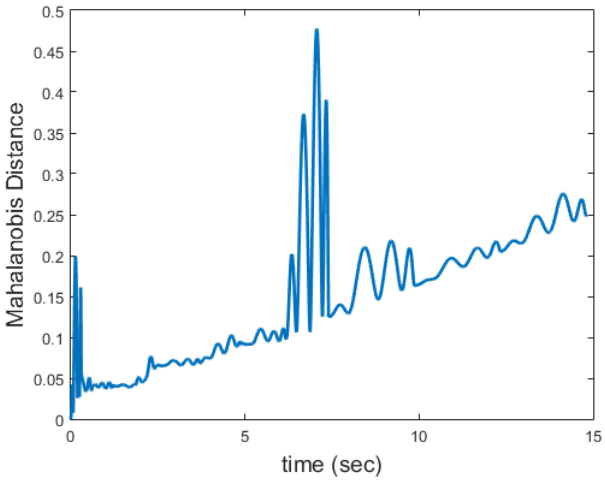

Fig. 7: The Mahalanobis Distance meets the threshold requirement of 0.5 after 5 mesh iterations.

The final interpolated and propagated heading are shown in Figure 8. While there are still small oscillations in the propagated heading, the interpolated value no longer exceeds the one standard deviation bounds, unlike what was seen prior to the first refinement in Figure 3. This is confirmation of the choice of MD as an appropriate error.

Finally, the overhead view of the trajectory in Figure 9 shows excellent agreement between the interpolated and propagated solutions, even after nearly 15 seconds of propagation without a measurement update. This indicates that the optimal

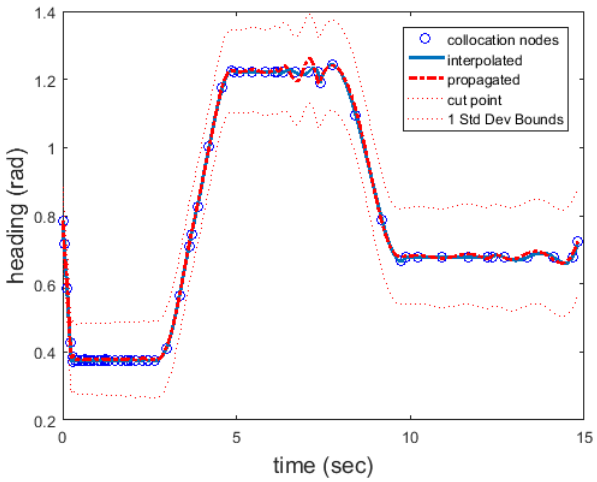

Fig. 8: The difference between the interpolated and propagated headings is relatively small after 5 mesh refinements. The small oscillations do not cause the MD to exceed the threshold.

control solution on the refined mesh is adequate for open-loop use. While it is possible that traditional methods of mesh refinement such as those described in [6]-[10] could produce a control solution on a mesh which would successfully propagate with low MD error, there is no guarantee. For this reason it is recommended that at a minimum the algorithm here is used as an additional check to other methods.

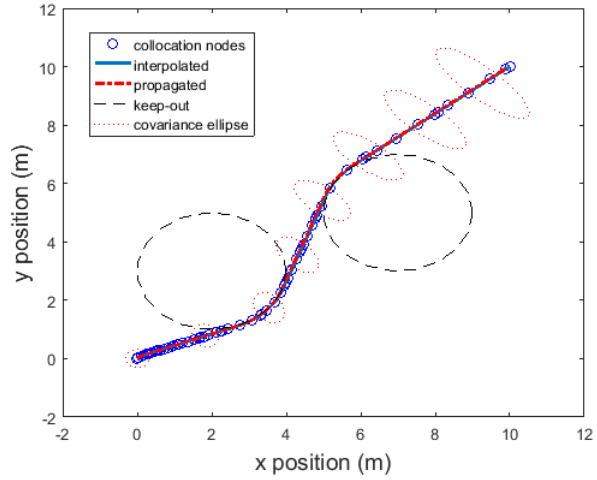

Fig. 9: No visible difference exists between the interpolated and propagated trajectories after 5 iterations of the mesh refinement algorithm.

V. Conclusion

While many methods exist for refining a mesh based purely on characteristics of the transcription method, the technique developed in this paper is the result of a holistic approach which looks at the eventual application of the optimal trajectory, and refines the mesh in order to enhance the applicability of the open-loop optimal control solution. The example problem provided here is simple, yet it highlights the need for appropriate mesh adaptation before using the optimal control solution as part of a larger controller. The Mahalanobis Distance, which serves as a gauge of relative error between the interpolated and propagated solutions, ties the trajectory generation and estimation modules of the overall control system together in a natural and convenient way. Although there is still a need for feedback control action due to unmodeled dynamics and disturbances, at least the control system is relieved of the need to also compensate for errors in the implementation of the optimal feedforward control.

References

[1] Philip E. Gill, Walter Murray, and Michael A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005.

[2] G. Elnagar, M.A. Kazemi, and M. Razzaghi. The pseudospectral Legendre method for discretizing optimal control problems. *Automatic Control, IEEE Transactions on*, 40(10):1793–1796, Oct 1995.

[3] A review of pseudospectral optimal control: From theory to flight. *Annual Reviews in Control*, 36:182–197, Apr 2012.

[4] J. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, second edition, 2010.

[5] J. P. Boyd. *Chebyshev and Fourier Spectral Methods: Second Revised Edition*. Dover Books on Mathematics. Dover Publications, 2001.

[6] Q. Gong, F. Fahroo, and M. Ross. Spectral algorithm for pseudospectral methods in optimal control. *Journal of Guidance, Control, and Dynamics*, 31(3):460–471, 208.

[7] Michael A. Patterson, William W. Hager, and Anil V. Rao. A ph mesh refinement method for optimal control. *Optimal Control Applications and Methods*, 36(4):398–421, 2015.

[8] Fengjin Liu, William W. Hager, and Anil V. Rao. Adaptive mesh refinement method for optimal control using nonsmoothness detection and mesh size reduction. *Journal of the Franklin Institute*, 352(10):4081 – 4106, 2015.

[9] Christopher L. Darby, William W. Hager, and Anil V. Rao. An hp-adaptive pseudospectral method for solving optimal control problems. *Optimal Control Applications and Methods*, 32(4):476–502, 2011.

[10] Christopher L. Darby, William W. Hager, and Anil V. Rao. Direct trajectory optimization using a variable low-order adaptive pseudospectral method. *Journal of Spacecraft and Rockets*, 48(3):433–445, May 2011.

[11] N. Bedrossian, S. Bhatt, M. Lammers, L. Nguyen, and Y. Zhang. First ever flight demonstration of zero propellant maneuver attitude control concept. In *AIAA Guidance, Navigation and Control Conference and Exhibit, Hilton Head, South Carolina, AIAA 2007-6734*, August 2007.

[12] S. M. Ross, R. G. Cobb, and W. P. Baker. Stochastic real-time optimal control for bearing-only trajectory planning. *International Journal of Micro Air Vehicles*, 6(1):1–27, 2014.

[13] Michael A. Patterson and Anil 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.*, 41(1):1:1–1:37, October 2014.

[14] Divya Garg, Michael A. Patterson, Camila Francolin, Christopher L. Darby, Geoffrey T. Huntington, William W. Hager, and Anil V. Rao. Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a radau pseudospectral method. *Computational Optimization and Applications*, 49(2):335–358, 2011.

[15] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

[16] J. J. LaViola. A comparison of unscented and extended kalman filtering for estimating quaternion motion. volume 3, pages 2435–2440, June 2003.

[17] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Prentice Hall, 2007.

X. Appendix C

Installation instructions for PSOPT onto a Raspberry Pi version 3 and an ODROID-C.

Note: The $ character signifies a command line argument. Just copy and paste without the $ into the command line.

Download the code for PSOPT v4 from the github page: https:github.com/PSOPT/psopt
Note: This version of the code hasn't been officially released at the current date. So no guarantees on performance, bugs, etc. However, it does compile with the most recent versions of IPOPT and other third party libraries. This code is likely to change in the future so it may be better to just obtain the code from another ODROID. If you're reading this, then you probably have a version that works with these instructions.

Prior to anything else, fully update the ODROID by entering the following commands into the console:

$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get dist-upgrade

Enter the main PSOPT folder, probably called 'psopt-master'. It contains a bash script written by the author of PSOPT called 'install-ubunut-16.04.sh'. This does not work on the ODROID without some modifications. Open this file with a text editor (I like gedit). Copy the following commands from the bash script into the command console and run them, one by one (or save them to a new bash script and run it). This will download a few third party libraries.

$ sudo apt-get -y install g++ gfortran f2c libf2c2-dev libf2c2 libblas-dev libopenblas-

base libopenblas-dev libblas3 libatlas-base-dev liblapack-dev liblapack3

$ cd $HOME/Downloads

$ wget –continue http:www.coin-or.org/download/source/Ipopt/Ipopt-3.12.3.tgz

$ cd $HOME

$ tar xzvf ./Downloads/Ipopt-3.12.3.tgz

$ cd $HOME/Ipopt-3.12.3/ThirdParty/Metis

$ ./get.Metis

$ cd $HOME/Ipopt-3.12.3/ThirdParty/Mumps

$ ./get.Mumps

$ cd $HOME/Ipopt-3.12.3

There is a problem with Metis, Mumps, and IPOPT not recognizing the current architecture. Update by starting in the Ipopt-3.12.3 folder and running:

$ wget -O config.guess 'http:git.savannah.gnu.org/gitweb/p=config.git;a=blob_plain;f=config.guess;hb=HEAD'

$ wget -O config.sub 'http:git.savannah.gnu.org/gitweb/?p=config.git;a=blob_plain;f=config.sub;hb=HEAD'

Copy the config.guess and config.sub files into the other folders:

$ cp guess.* /ThirdParty/Metis/.

$ cp guess.* /ThirdParty/Mumps/.

$ cp guess.* /Ipopt/.

You can now run the next command in the Ipopt-3.12.3 folder to configure IPOPT

$ ./configure –enable-static coin/_skip/_warn_cxxflags=yes

You should see a message saying 'Main configuration of Ipopt successful'. The next step would have you run 'make -j', which uses all four processors to make IPOPT. For some reason this doesn't work, so just run (this takes a while):

$ make

If this works, 15 minutes later you can run the following, which will require administrative approval. The default password for the ODROID is 'odroid':

$ sudo make install

Run the commands all the way through:

$ cd $HOME/Downloads
$ wget –continue www.coin-or.org/download/source/ADOL-C/ADOL-C-2.5.2.tgz
$ cd $HOME
$ tar zxvf ./Downloads/ADOL-C-2.5.2.tgz
$ cd $HOME/ADOL-C-2.5.2
$ mkdir ./ThirdParty
$ cd ./ThirdParty
$ wget –continue http:cscapes.cs.purdue.edu/download/ColPack/ColPack-1.0.9.tar.gz
$ tar zxvf ColPack-1.0.9.tar.gz
$ mv ColPack-1.0.9 ColPack
$ cd ColPack

Again, the config.guess and config.sub files are out of date. Grab the new ones and paste them:

$ cp ../../Ipopt-3.12.3/config.* .

Now run the following:

$ ./configure
$ make
$ sudo make install
$ sudo cp /usr/local/lib/libCol* /usr/lib
$ cd $HOME/ADOL-C-2.5.2
$ ./configure –enable-sparse –with-colpack=$HOME/ADOL-C-2.5.2/ThirdParty/ColPack
$ make
$ make install
$ sudo cp $HOME/adolc_base/lib64/*.a /usr/lib
$ sudo cp -r $HOME/adolc_base/include/* /usr/include/

Next you'll install an optional package, PDF lite, which the author uses to make plots from gnuplot

$ cd $HOME/Downloads
$ wget –continue http:www.pdflib.com/binaries/PDFlib/705/PDFlib-Lite-7.0.5p3.tar.gz
$ tar zxvf PDFlib-Lite-7.0.5p3.tar.gz
$ cd PDFlib-Lite-7.0.5p3

Again, we'll have to copy the geuss.conf and guess.sub files

$ cp ../../Ipopt-3.12.3/config.* .

$ cp ../../Ipopt-3.12.3/config.* config/.

Then continue with the install:

$ ./configure

$ make; sudo make install

$ sudo ldconfig

Install gnuplot:

$ cd $HOME/Downloads

$ wget –continue

http:sourceforge.net/projects/gnuplot/files/gnuplot/4.2.2/gnuplot-4.2.2.tar.gz/download

$ mv download gnuplot-4.2.2.tar.gz

$ tar zxvf gnuplot-4.2.2.tar.gz

$ sudo apt-get -y install libx11-dev libxt-dev libgd2-xpm-dev libreadline6-dev

$ cd gnuplot-4.2.2

$ ./configure -with-readline=gnu -without-tutorial

$ make

$ sudo make install

Finally, download and install PSOPT. You may skip the PSOPT download part

if the code is updated and you simply want to use the current code.

$ cd $HOME

$ wget –continue https:github.com/PSOPT/psopt/archive/master.zip

$ unzip master.zip

$ mv master.zip $HOME/Downloads

$ cd $HOME/psopt-master

$ wget –continue http:faculty.cse.tamu.edu/davis/SuiteSparse/SuiteSparse-4.4.3.tar.gz

$ tar zxvf SuiteSparse-4.4.3.tar.gz

$ cd $HOME/psopt-master

$ wget –continue http:www.stanford.edu/group/SOL/software/lusol/lusol.zip

$ unzip lusol.zip

$ cd $HOME/psopt-master

$ make all

That's it!

Bibliography

[1] Air Force Research Laboratory. Air Force Research Laboratory Autonomy Science and Technology Strategy. http://defenseinnovationmarketplace.mil/resources/AFRLAutonomyStrategy-DistroA.pdf.

[2] John R Boyd, Thomas P Christie, and James E Gibson. Energy maneuverability (u). *Air Proving Ground Center Report APGC-TR-66-4 Vol*, 1, 1966.

[3] T. Basar. Relaxation techniques and asynchronous algorithms for on-line computation of noncooperative equilibria. In *Decision and Control, 1987. 26th IEEE Conference on*, volume 26, pages 275–280, Dec 1987.

[4] D.E. Kirk. *Optimal Control Theory: An Introduction*. Dover Books on Electrical Engineering Series. Dover Publications, 2004.

[5] J. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, second edition, 2010.

[6] Philip E. Gill, Walter Murray, and Michael A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005.

[7] MATLAB. *version 8.3.0 (R2014a)*. The MathWorks Inc., Natick, Massachusetts, 2014.

[8] Andreas Wachter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[9] S.M. Ross, R.G. Cobb, and W.P. Baker. Stochastic real-time optimal control for bearing-only trajectory planning. *International Journal of Micro Air Vehicles*, 6(1):1–27, 2014.

[10] Nathan E. Smith, Richard Cobb, Scott J. Pierce, and Vincent Raska. *Optimal Collision Avoidance Trajectories via Direct Orthogonal Collocation for Unmanned/Remotely Piloted Aircraft Sense and Avoid Operations*. AIAA SciTech. American Institute of Aeronautics and Astronautics, Jan 2014.

[11] Angela W. Suplisson, Richard Cobb, William Baker, and David Jacques. *An Optimal Control Approach to Aircraft Automatic Ground Collision Avoidance*. AIAA SciTech. American Institute of Aeronautics and Astronautics, Jan 2015.

[12] R. Burden, J. Faires, and A. Burden. *Numerical Analysis*. Cengage Learning, 9 edition, 2011.

[13] J.P. Boyd. *Chebyshev and Fourier Spectral Methods: Second Revised Edition*. Dover Books on Mathematics. Dover Publications, 2001.

[14] G. Elnagar, M.A. Kazemi, and M. Razzaghi. The pseudospectral Legendre method for discretizing optimal control problems. *Automatic Control, IEEE Transactions on*, 40(10):1793–1796, Oct 1995.

[15] Divya Garg, Michael Patterson, William W. Hager, Anil V. Rao, David A. Benson, and Geoffrey T. Huntington. A unified framework for the numerical solution of optimal control problems using pseudospectral methods. *Automatica*, 46(11):1843 – 1851, 2010.

[16] L. Trefethen. *Spectral Methods in MATLAB*. Society for Industrial and Applied Mathematics, 2000.

[17] Divya Garg, Michael A. Patterson, Camila Francolin, Christopher L. Darby, Geoffrey T. Huntington, William W. Hager, and Anil V. Rao. Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a radau pseudospectral method. *Computational Optimization and Applications*, 49(2):335–358, 2011.

[18] Michael A. Patterson and Anil 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.*, 41(1):1:1–1:37, October 2014.

[19] V.M. Becerra. Solving complex optimal control problems at no cost with PSOPT. In *Computer-Aided Control System Design (CACSD), 2010 IEEE International Symposium on*, Sept 2010.

[20] Fariba Fahroo and I. Michael Ross. *Advances in Pseudospectral Methods for Optimal Control*. Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, Aug 2008.

[21] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley, 2006.

[22] Michael A. Patterson, William W. Hager, and Anil V. Rao. A ph mesh refinement method for optimal control. *Optimal Control Applications and Methods*, 36(4):398–421, 2015.

[23] Christopher L. Darby, William W. Hager, and Anil V. Rao. An hp-adaptive pseudospectral method for solving optimal control problems. *Optimal Control Applications and Methods*, 32(4):476–502, 2011.

[24] Fengjin Liu, William W. Hager, and Anil V. Rao. Adaptive mesh refinement method for optimal control using nonsmoothness detection and mesh size reduction. *Journal of the Franklin Institute*, 352(10):4081 – 4106, 2015.

[25] Michael A. Patterson and Anil Rao. Exploiting sparsity in direct collocation pseudospectral methods for solving optimal control problems. *Journal of Spacecraft and Rockets*, 49(2):354–377, Mar 2012.

[26] Q. Gong, I. Ross, W. Kang, and F. Fahroo. On the pseudospectral covector mapping theorem for nonlinear optimal control. In *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006.

[27] F. Fahroo and I. Ross. Costate estimation by a legendre pseudospectral method. *Journal of Guidance, Control, and Dynamics*, 24(2):270–277, Mar 2001.

[28] P. Zarchan. *Tactical and strategic missile guidance*. Progress in Astronautics and Aeronautics. American Institute of Aeronautics and Astronautics, 2007.

[29] B. Gunston. *An illustrated guide to modern airborne missiles*. Salamander Books, 1983.

[30] M. W. Fossier. The development of radar homing missiles. *Journal of Guidance, Control, and Dynamics*, 7(6):641–651, Nov 1984.

[31] R. Yanushevsky. *Modern Missile Guidance*. CRC Press, 2007.

[32] Robert M Olberg. Visual control of prey-capture flight in dragonflies. *Current Opinion in Neurobiology*, 22(2):267 – 271, 2012. Neuroethology.

[33] Kaushik Ghose, Timothy K Horiuchi, P. S Krishnaprasad, and Cynthia F Moss. Echolocating bats use a nearly time-optimal strategy to intercept prey. *PLoS Biol*, 4(5):e108, 04 2006.

[34] Dennis M. Shaffer, Scott M. Krauchunas, Marianna Eddy, and Michael K. McBeath. How dogs navigate to catch frisbees. *Psychological Science*, 15(7):437–441, 2004.

[35] Dominique Martinez. Klinotaxis as a basic form of navigation. *Front Behav Neurosci*, 8:275, Aug 2014. 25177280[pmid].

[36] McBeath M., Shaffer D., and Kaiser M. How baseball outfielders determine where to run to catch fly balls. *Science*, 268:569–573, April 1995.

[37] Palumbo N., Bauwkamp R., and Lloyd J. Modern homing missile guidance theory and techniques. *Johns Hopkins APL Technical Digest*, 29(1), 2010.

[38] M. Guelman. A qualitative study of proportional navigation. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-7(4):637–643, July 1971.

[39] Ciann-Dong Yang and Chi-Ching Yang. A unified approach to proportional navigation. *Aerospace and Electronic Systems, IEEE Transactions on*, 33(2):557–567, April 1997.

[40] Pin-Jar Yuan and Jeng-Shing Chern. Solutions of true proportional navigation for maneuvering and nonmaneuvering targets. *Journal of Guidance, Control, and Dynamics*, 15(1):268–271, Jan 1992.

[41] Stephen A. Murtaugh and Harry E. Criel. Fundamentals of proportional navigation. *Spectrum, IEEE*, 3(12):75–85, Dec 1966.

[42] Ciann-Dong Yang, Fang-Bo Yeh, and Jen-Heng Chen. The closed-form solution of generalized proportional navigation. *Journal of Guidance, Control, and Dynamics*, 10(2):216–218, Mar 1987.

[43] Pin-Jar Yuan and Jeng-Shing Chern. Ideal proportional navigation. *Journal of Guidance, Control, and Dynamics*, 15(5):1161–1165, Sep 1992.

[44] Janne Karelahti, Kai Virtanen, and Tuomas Raivio. Near-optimal missile avoidance trajectories via receding horizon control. *Journal of Guidance, Control, and Dynamics*, 30(5):1287–1298, Sep 2007.

[45] Fumiaki Imado and Sachio Uehara. High-g barrel roll maneuvers against proportional navigation from optimal control viewpoint. *Journal of Guidance, Control, and Dynamics*, 21(6):876–881, Nov 1998.

[46] Vincent Lam. *Time-to-Go Estimate for Missile Guidance*. Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, Aug 2005.

[47] Min-Jea Tahk, Chang-Kyung Ryoo, and Hangju Cho. Recursive time-to-go estimation for homing guidance missiles. *Aerospace and Electronic Systems, IEEE Transactions on*, 38(1):13–24, Jan 2002.

[48] N. Dhananjay and D. Ghose. Accurate time-to-go estimation for proportional navigation guidance. *Journal of Guidance, Control, and Dynamics*, 37(4):1378–1383, Mar 2014.

[49] David Hull and Jerry Radke. *Time-to-go prediction for a homing missile based on minimum-time trajectories*. Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, Aug 1988.

[50] G. K. F. Lee. Estimation of the time-to-go parameter for air-to-air missiles. *Journal of Guidance, Control, and Dynamics*, 8(2):262–266, Mar 1985.

[51] R. E. Ball. *The Fundamentals of Aircraft Combat Survivability Analysis and Design, 2nd Edition*. American Institute of Aeronautics and Astronautics, 2003.

[52] R. Isaacs. *Differential Games*. John Wiley and Sons, Inc., New York, 1965.

[53] Gerald M. Anderson. Comparison of optimal control and differential game intercept missile guidance laws. *Journal of Guidance, Control, and Dynamics*, 4(2):109–115, Mar 1981.

[54] Shaul Gutman. On optimal guidance for homing missiles. *Journal of Guidance, Control, and Dynamics*, 2(4):296–300, Jul 1979.

[55] J. Shinar and D. Steinberg. Analysis of optimal evasive maneuvers based on a linearized two-dimensional kinematic model. *Journal of Aircraft*, 14, Aug 1978.

[56] J. Shinar. Analysis of three-dimensional optimal evasion with linearized kinematics. *Guidance, Navigation, and Control and Co-located Conferences*, Aug 1978.

[57] Leena Singh. Autonomous missile avoidance using nonlinear model predictive control. *Guidance, Navigation, and Control and Co-located Conferences*, Aug 2004.

[58] Martin Weiss and Tal Shima. Minimum effort pursuit/evasion guidance with specified miss distance. *Journal of Guidance, Control, and Dynamics*, pages 1–11, Jan 2016.

[59] Janne Karelahti, Kai Virtanen, and Tuomas Raivio. Game optimal support time of a medium range air-to-air missile. *Journal of Guidance, Control, and Dynamics*, 29(5):1061–1069, Sep 2006.

[60] Shaw Y. Ong and Bion L. Pierson. Optimal planar evasive aircraft maneuvers against proportional navigation missiles. *Journal of Guidance, Control, and Dynamics*, 19(6):1210–1215, Nov 1996.

[61] Fumiaki Imado and Susumu Miwa. Fighter evasive maneuvers against proportional navigation missile. *Journal of Guidance, Control, and Dynamics*, 23(11):825–830, Nov 1986.

[62] F. Imado and S. Miwa. *The optimal evasive maneuver of a fighter against proportional navigation missiles*. Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, Aug 1983.

[63] J.A. Kaplan, A.R Chappell, and J.W. McManus. The analysis of a generic air-to-air missile simulation model. Technical Report 109057, NASA, June 1994.

[64] E.S. Rutowski. Energy approach to the general aircraft performance problem. *Journal of the Aeronautical Sciences*, 21(3):187–195, Mar 1954.

[65] A.E. Bryson, M.N. Desai, and W.C. Hoffman. Energy-state approximation in performance optimization of a supersonic aircraft. *Journal of Aircraft*, 6(6):481–488, December 1969.

[66] J. Shinar and R. Tabak. New results in optimal missile avoidance analysis. *Journal of Guidance, Control, and Dynamics*, 17(5):897–902, September 1994.

[67] P. Menon and M. Briggs. A midcourse guidance law for air-to-air missiles. *AIAA Guidance, Navigation, and Control and Co-located Conferences*, Aug 1987.

[68] Fumiaki Imado, Takeshi Kuroda, and Susumu Miwa. Optimal midcourse guidance for medium-range air-to-air missiles. *Journal of Guidance, Control, and Dynamics*, 13(4):603–608, Jul 1990.

[69] Sergey Rubinsky and Shaul Gutman. Three-player pursuit and evasion conflict. *Journal of Guidance, Control, and Dynamics*, 37(1):98–110, Dec 2013.

[70] Eloy Garcia, David W. Casbeer, and Meir Pachter. Cooperative strategies for optimal aircraft defense from an attacking missile. *Journal of Guidance, Control, and Dynamics*, 38(8):1510–1520, Apr 2015.

[71] B. J. Roadruck. Counter weapon control. Thesis, Air Force Institute of Technology, 2015.

[72] T. Basar and G.J. Olsder. *Dynamic Noncooperative Game Theory*. Society for Industrial and Applied Mathematics, 1998.

[73] L.J. Ratliff, S.A. Burden, and S.S. Sastry. Characterization and computation of local nash equilibria in continuous games. In *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*, pages 917–924, Oct 2013.

[74] John Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951.

[75] Alberto Bressan. Noncooperative differential games. a tutorial, 2010. Department of Mathematics, Penn State University, Course Notes.

[76] A. Friedman. *Differential Games*. John Wiley and Sons, Inc., New York, 1971.

[77] D. M. Salmon. Relaxation techniques and asynchronous algorithms for on-line computation of noncooperative equilibria. *IEEE Transactions on Automatic Control*, 14(5):482–488, Oct 1969.

[78] L. Meier. A new technique for solving pursuit-evasion differential games. *IEEE Transactions on Automatic Control*, 14(4):352–359, Aug 1969.

[79] A. W. Merz. The homicidal chauffeur. *AIAA Journal*, 12(3):259–260, Mar 1974.

[80] Y. Ho, A. Bryson, and S. Baron. Differential games and optimal pursuit-evasion strategies. *IEEE Transactions on Automatic Control*, 10(4):385–389, Oct 1965.

[81] G. T. Rublein. On pursuit with curvature constraints. *SIAM Journal on Control*, 10(1):37–39, 1972.

[82] J. Dolezal. A gradient-type algorithm for the numerical solution of two-player zero-sum differential game problems. *Kybernetika*, 14(6), 1978.

[83] Tuomas Raivio. Capture set computation of an optimally guided missile. *Journal of Guidance, Control, and Dynamics*, 24(6):1167–1175, Nov 2001.

[84] H. Ehtamo and Tuomas Raivio. On applied nonlinear and bilevel programming for pursuit-evasion games. *Journal of Optimization Theory and Applications*, 108(1):65–96, January 2001.

[85] Kazuhiro Horie and Bruce A. Conway. Optimal fighter pursuit-evasion maneuvers found via two-sided optimization. *Journal of Guidance, Control, and Dynamics*, 29(1):105–112, Jan 2006.

[86] G. Papavassilopoulos and J. Cruz. Nonclassical control problems and stackelberg games. *IEEE Transactions on Automatic Control*, 24(2):155–166, Apr 1979.

[87] Mauro Pontani and Bruce A. Conway. Optimal interception of evasive missile warheads: Numerical solution of the differential game. *Journal of Guidance, Control, and Dynamics*, 31(4):1111–1122, 2008.

[88] Mauro Pontani and Bruce A. Conway. Numerical solution of the three-dimensional orbital pursuit-evasion game. *Journal of Guidance, Control, and Dynamics*, 32(2):474–487, 2009.

[89] Songtao Sun, Qiuhua Zhang, Ryan Loxton, and Bin Li. Numerical solution of a pursuit-evasion differential game involving two spacecraft in low earth orbit. *Journal of Industrial and Management Optimization*, 11(4), 2015.

[90] K Horie. *Collocation with Nonlinear Programming for Two-Sided Flight Path Optimization*. Dissertation, University of Illinois at Urbana-Champaign, 2002.

[91] Kazuhiro Horie and Bruce A. Conway. GA pre-processing for numerical solution of differential games problems. *Journal of Guidance, Control, and Dynamics*, 27(6):1075–1078, 2004.

[92] William Hafer, Helen Reed, James Turner, and Khan Pham. Sensitivity methods applied to orbital pursuit evasion. *Journal of Guidance, Control, and Dynamics*, 38(6):118–1126, 2015.

[93] A.W. Starr. Computation of nash equilibria for non-linear differential games. In *Proceedings of the First International Conference on teh Theory and Application of Dynamic Games*, pages IV–13–IV–18, Sep 1969.

[94] Shu Li and Tamer Başar. Distributed algorithms for the computation of noncooperative equilibria. *Automatica*, 23(4):523–533, 1987.

[95] S. Uryas'ev and R. Y. Rubinstein. On relaxation algorithms in computation of noncooperative equilibria. *IEEE Transactions on Automatic Control*, 39(6):1263–1267, Jun 1994.

[96] Changsu Park and Min-Jea Tahk. A coevolutionary minimax solver and its application to autopilot design. In *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Boston, Massachussetts, AIAA-98-4197*, 1998.

[97] D. Shen, B. Jia, G. Chen, K. Pham, and E. Blasch. Space based sensor management strategies based on informational uncertainty pursuit-evasion games. In *Aerospace and Electronics Conference (NAECON)*, 2015.

[98] Tuomas Raivio and Harri Ehtamo. Visual aircraft identification as a pursuit-evasion game. *Journal of Guidance, Control, and Dynamics*, 23(4), 2000.

[99] J. Mattingley, Yang Wang, and S. Boyd. Receding horizon control. *Control Systems, IEEE*, 31(3):52–65, June 2011.

[100] Steven M. Ross, Richard G. Cobb, William P. Baker, and Frederick G. Harmon. Implementation lessons and pitfalls for real-time optimal control with stochastic systems. *Optimal Control Applications and Methods*, 36(2):198–217, 2015.

[101] Jon Strizzi, I Michael Ross, and Fariba Fahroo. Towards real-time computation of optimal controls for nonlinear systems. In *AIAA Guidance, Navigation, and Control Conference, Monterey, CA, USA*, 2002.

[102] I Michael Ross, Pooya Sekhavat, Andrew Fleming, Qi Gong, and W Wei Kang. Pseudospectral feedback control: foundations, examples and experimental results. In *Proceedings of the 2006 AIAA Guidance, Navigation, and Control Conference*, 2006.

[103] Michael A Hurni, Pooya Sekhavat, and I Michael Ross. Autonomous trajectory planning using real-time information updates. In *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, Hawaii, AIAA-2008-6305*, 2008.

[104] S. M. Ross. *Stochastic real-time optimal control: A pseudospectral approach for bearing-only trajectory optimization.* Dissertation, Air Force Institute of Technology, 2011.

[105] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

[106] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, Mar 2004.

[107] E.A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158, 2000.

[108] R. Van Der Merwe and E.A. Wan. The square-root unscented kalman filter for state and parameter-estimation. In *Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001 IEEE International Conference on*, volume 6, pages 3461–3464 vol.6, 2001.

[109] S.J. Julier. The scaled unscented transformation. In *American Control Conference, 2002. Proceedings of the 2002*, volume 6, pages 4555–4559 vol.6, 2002.

[110] Nathan E. Smith, Richard Cobb, Scott Pierce, and Vincent Raska. *Optimal Collision Avoidance Trajectories for Unmanned/Remotely Piloted Aircraft.* Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, Aug 2013.

[111] Pierre Del Moral. Nonlinear filtering: Interacting particle resolution. *Comptes Rendus de l'Academie des Sciences - Series I - Mathematics*, 325(6):653 – 658, 1997.

[112] Angela W Suplisson. *Optimal Recovery Trajectories for Automatic Ground Collision Avoidance Systems (Auto GCAS).* Dissertation, Air Force Institute of Technology, 2015.

[113] Standard atmosphere - tables and data for altittudes to 65,800 feet. Technical Report Report 1235, National Advisory Committee for Aeronautics, January 1955.

[114] Fumiaki Imado and Takeshi Kuroda. Family of local solutions in a missile-aircraft differential game. *Journal of Guidance, Control, and Dynamics*, 34(2):583–591, March 2011.

[115] D. Hull. *Optimal Control Theory for Applications.* Mechanical Engineering Series, SPringer-Verlag, New York, 2003.

[116] G. Knowles. *An Introduction to Applied Optimal Control.* Mathematics In Science and Engineering, Volume 159, Academic Press, Inc, 1981.

[117] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. *The 14th International Conference on Artificial Intelligence and Statistics*, 15(106), 2011.

[118] R. Johnson and D. Wichern. *Applied Multivariate Statistical Analysis, 6th Edition.* Pearson Prentice Hall, 2007.

Vita

Major Ryan Carr has been a PhD student in optimal control in the Department of Aeronautics and Astronautics at the Air Force Institute of Technology since August 2014. His 12 years in the Air Force include jobs working for the Air Force Research Laboratory, the Air Force Office of Scientific Research, and the Air Force Flight Test Center.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
14–09–2017	Doctoral Dissertation	Aug 2014 – Sep 2017

4. TITLE AND SUBTITLE

Optimal Control Methods
for Missile Evasion

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Carr, Ryan W., Major, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENY-DS-17-S-055

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

Amy Burns
AFRL/RQQC, Control Systems Branch
2130 8th Street
Wright Patterson AFB, OH 45433-7103
(937) 938-4603, amy.burns.3@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RQQC

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

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

14. ABSTRACT Optimal control theory is applied to the study of missile evasion, particularly in the case of a single pursuing missile versus a single evading aircraft. It is proposed to divide the evasion problem into two phases, where the primary considerations are energy and maneuverability, respectively. Two problems are proposed as surrogates for the energy phase; a fixed final time problem with free final state and a free final time problem with fixed final state. These two optimal control problems are studied under several different scenarios regarding assumptions about the pursuer. First, a suboptimal control strategy, proportional navigation, is used for the pursuer. Second, it is assumed that the pursuer acts optimally, requiring the solution of a two-sided optimal control problem, otherwise known as a differential game. The resulting trajectory is known as a minimax, and it can be shown that it accounts for uncertainty in the pursuer's control strategy. Finally, a pursuer whose motion and state are uncertain is studied in the context of Receding Horizon Control and Real Time Optimal Control. The results highlight how updating the optimal control trajectory reduces the uncertainty in the resulting miss distance.

15. SUBJECT TERMS

Missile Evasion, Optimal Control, Minimax, Pursuit-Evasion, Guaranteed-Cost Control

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Richard G. Cobb (ENY)
U	U	U	UU	214	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 x4559 Richard.Cobb@afit.edu