RECEDING HORIZON CONTROL FOR UNCERTAIN PURSUIT-EVASION GAMES

A Thesis

by

BRIAN SCOTT JANISCH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	John E. Hurtado
Committee Members,	S. Rao Vadali
	John Valasek
	Shankar Bhattacharyya
Head of Department,	Rodney D. W. Bowersox

August 2016

Major Subject: Aerospace Engineering

ABSTRACT

A robust technique for handling parameter and strategy uncertainty in a pursuit-evasion framework is developed. The method is a receding horizon controller valid for problem classes with singularly perturbed trajectories that approximates the optimal feedback solution with small loss in optimality. The receding horizon method is used to ensure the controller is robust to incorrect or extraneous information about an opposing player's dynamics or strategy. A simple analytic pursuit-evasion game motivates the method by demonstrating that the receding horizon solution closely approximates the optimal solution and may be solved much faster. Simulations of a nonlinear game show that the receding horizon controller is especially useful when it is unknown whether the opposing player is performing an active or passive maneuver. In several cases, the receding horizon controller is shown to become more effective than a game-optimal controller acting with an incorrect strategy estimate. The major limitation of the technique for a nonlinear system is the expensive solution time; therefore, the optimal control problem is translated to a nonlinear programming problem and the test cases are repeated. Finally, the test cases are run on hardware to validate the method for real-time practical operation.

The singular-perturbation algorithm applied herein is valid only for a small subset of all pursuit and evasion games. Nonetheless, the methods developed here can in theory be used for any generic game scenario, given that sufficient computing power is available to find the numerical solutions.

To my parents and role models, Mom and Dad.

ACKNOWLEDGEMENTS

First and foremost I would like to thank my advisor, Dr. John E. Hurtado, for providing me with valuable direction, insight, and vision during my time here at Texas A&M University. Working together has been especially rewarding for me, as I got to learn from someone who has worked across several sectors and excels as an educator. I also owe thanks to the other excellent educators and researchers serving on my defense committee - Dr. Srinivas Rao Vadali, Dr. John Valasek, and Dr. Shankar Bhattacharrya. Discussions and courses taken with these professors helped mold my interest and spark ideas about my thesis, and several conversations with each contributed to the direction my work went. I owe additional thanks to everyone at the LASR Lab for their help and for putting up with my terrible puns.

I would like to acknowledge my research sponsors at Sandia National Laboratories for funding me during the duration of my study, and for exposing me to additional interesting and rewarding research and work that I may not have otherwise seen. Dr. Kevin Brink at the Air Force Research Lab was also instrumental in helping me get started with pursuit-evasion theory, and especially with different types of uncertainty and strategies in pursuit-evasion games. Working with him at the REEF lab was a great way to get some early hardware experience to build from as my work developed further.

Finally and most importantly, I owe the most thanks to my family and girlfriend Taylor. Words cannot capture the encouragement and support they have shown not only during my time at A&M, but all throughout my education and toward my

pursuit of a second degree. The care packages, cards, visits, support, and love are really what got me through the times I felt overwhelmed, isolated, or defeated, and are what bolstered me higher any time I felt joy or accomplishment. Thank you from the bottom of my heart.

NOMENCLATURE

ACADO	Automatic Control and Dynamic Optimization
CPU	Central Processing Unit
HC	Homicidal Chauffuer
HWIL	Hardware-in-the-Loop
LASR	Land Air and Space Robotics (laboratory)
LO	Loss in Optimality
LOS	Line of Sight
LSQ	Least-Squares
MC	Monte Carlo
MoCap	Motion Capture
NASA	National Aeronautics and Space Administration
NLP	Nonlinear Programming
OCP	Optimal Control Problem
PE	Pursuit-Evasion
$\mathbb{R}$	Set of Real Numbers
RHC	Receding Horizon Control
ROS	Robot Operating Software
SPT	Singular(ly) Perturbed Trajectory
SQP	Sequential Quadratic Programming
ZOH	Zero Order Hold

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

I.A. Motivation

Differential game theory and pursuit-evasion (PE) game theory have existed since 1951, when Rufus Isaacs published his fundamental work for RAND corporation [1]. Differential game theory is a special class of optimal control that employs a game-theoretic approach. It becomes differential theory when the modeled systems have dynamics governed by differential equations. Pursuit-Evasion problems are differential games where some pursuer attempts to intercept or capture an evader by minimizing a chosen performance index, or cost function, while the evader acts to prevent that outcome. A zero-sum game is one in which a gain in performance for one player translates to an equivalent loss in performance for the other.

Traditionally, pursuit-evasion theory considers a one-to-one player mapping with a zero-sum formulation, but has expanded to cover many scenarios like cooperative capture and nonzero-sum solutions [2, 3]. Today PE theory can be used to model interactions between competing aerospace vehicles, missile defense systems, or satellite relative motion, capture, and rendezvous. The standard PE approach to these problems assumes a deterministic, idealized encounter; however, we know the reality to be susceptible to disturbances, sensor noise, and model, parameter, and strategy uncertainty.

For minimum-time PE games, the 'final-time-free' optimal control is found over an infinite horizon with terminal constraints. An excellent treatment of optimal con-

trol theory is available in [4]. Most nonlinear systems require numerical minimization to solve for the optimal control over the infinite horizon; analytic solutions do not often exist. One problem inherent to solving the numerical infinite-horizon optimal control is the computation complexity. Many solvers use interior point or sequential quadratic programming (SQP) approaches to numerically solve for the control law that minimizes the chosen cost function. These solutions may recover the optimal control and trajectory, but often after heavy computation that cannot be reliably run on hardware in real-time. Because of these limitations to the general pursuit-evasion solution, a better approach is required. The solution presented herein is a receding horizon control (RHC) that approximates the optimal solution with minimal loss in optimality (LO).

Receding horizon control is an approach based on the idea of solving an optimal control problem repeatedly over short fixed horizons. During each fixed horizon period, a feedback approach can be implemented which corrects for errors from the previous finite period. Two important parameters of the receding horizon approach are the planning horizon length t_H and the execution time t_E. At the start of each horizon, the optimal control over the interval $[t_0, t_0 + t_H]$ is determined, and the first term of the control sequence $\boldsymbol{u} = \{u_{t_0}, u_{t_0+1}, ...u_{t_0+t_H}\}$ is applied and held for the duration of the execution time (execution time is synonymous with control bandwidth here). This is depicted in Fig. I.1. This is by no means the first use of RHC to solve PE games. In [5], it is used extensively to solve nonlinear PE games of a certain model structure. In [6] and [7], a robust RHC, or model predictive controller (MPC), control law is developed that ensures successful game completion

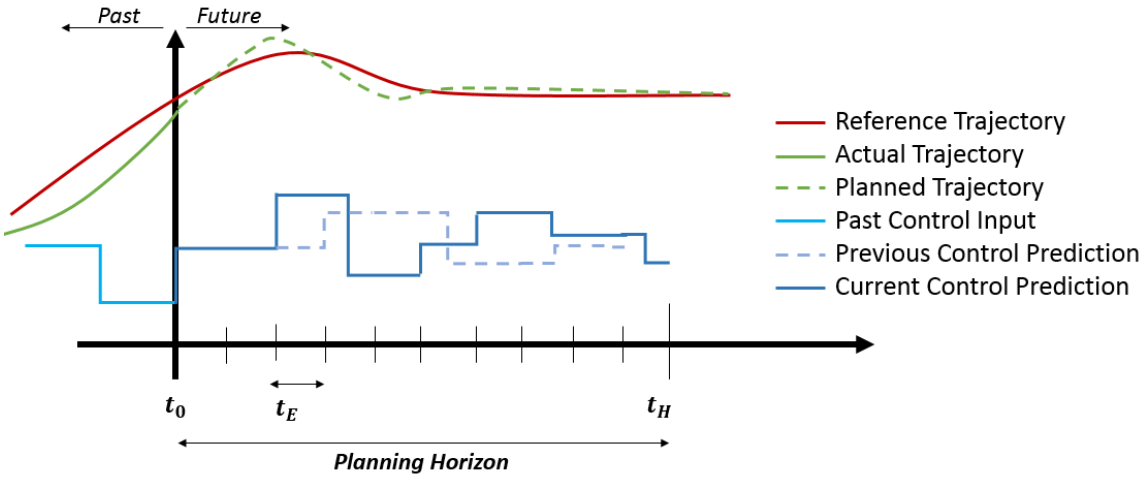


Figure I.1: Graphic of a typical RHC scheme

of nonlinear systems by playing conservatively and assuming a worst-case strategy. A real-time solver is written and validated for this approach. In [8] and [9], linear model PE problems are solved by utilizing behavior learning laws to estimate the strategy and parameters of the opposing player's model. All are valuable contributions to the field, but the goal of this work is to apply and validate a control structure that not only (a) closely mimics the optimal solution and (b) is valid for nonlinear systems and control but also (c) is not overly conservative, and (d) can be run online in real-time.

I.B. Outline

The structure of this thesis is organized as follows: in Chapter II a simple pursuit-evasion problem with a closed-form analytic solution is solved. Properties of both the problem and its solution give insight to the benefits and limitations of the receding horizon approach, and motivate RHC use for more general nonlinear and

complex examples with uncertainty. In Chapter III, the homicidal chauffeur problem is used as the dynamic model for higher-fidelity investigation and validation of the receding horizon approach. A control is developed to return nearly the same solution as the numerical optimal control problem to within one percent difference. It is both simulated in MATLAB and solved in real-time, and proper control approximations and assumptions are discussed. Chapter IV details the implementation of the controller on actual hardware and the accompanying experimental results. For this, the Land, Air, and Space Robotics (LASR) laboratory is utilized. The LASR lab has a central focus toward the development and evaluation of guidance, navigation, and control techniques on a variety of hardware-in-the-loop (HWIL) platforms and is well equipped for this implementation and experimentation. Finally, Chapter V contains a summary of all results and concludes with lessons learned and potential future work and applications of this method.

CHAPTER II

MOTIVATING PROBLEM: ONE-DIMENSIONAL RENDEZVOUS

II.A. Problem Statement

To introduce and motivate the use of the RHC method for pursuit-evasion games with strategy uncertainty, a simple one-dimensional PE game is solved. Solution of the game using both a closed-form feedback and a RHC method show that the methods' results differ only very slightly. By introducing an uncertainty or dynamic change to the problem, it is shown that the RHC method closely approximates the optimal feedback solution but never exceeds its performance. It is also shown that the RHC algorithm can trade accuracy for speed and thus is a good candidate for systems with numerical solutions that must be solved in real-time. The kinematic model chosen for the simple example is of two double integrators, where only one has constrained acceleration-level control. In a one-dimensional space, the goal of the game is to match speed and position along an infinite line. This is shown in Fig. II.1. The relative dynamics are written with respect to the pursuer such that a negative relative position means that $x_1 < 0$ and the evader is leading the pursuer. The pursuer's objective J_p is to minimize rendezvous time t_f, where the subscript f

Figure II.1: Double-integrator PE game illustration

denotes the final time and rendezvous is defined as $(x_1(t_f), x_2(t_f)) = (0, 0)$.

$$\dot{x}_1 = (v_p - v_e) = x_2 \tag{2.1}$$

$$\dot{x}_2 = (u - 0) = u \tag{2.2}$$

$$-1 \leq u \leq 1 \tag{2.3}$$

$$J_p = (t_f - t_0) = \int_{t_0}^{t_f} dt \tag{2.4}$$

II.B. Analytical Solution

The double integrator problem is a classic literature example and its optimal control solution can be found in several well-known texts [4, 10]. In the case where the control is constrained to some admissible region $u \in \mathcal{U}$, as is often physically realistic, Pontryagin's Minimum Principle defines the optimal value of u that minimizes the Hamiltonian. For this class of linear minimum-time problems, the optimal control is a bang-bang sequence [10]. The closed-form solution to the double integrator problem presented here is a switching function dependent on the value of x_1 [4].

When $x_1 > 0$

$$u = \begin{cases} -1, & x_1 \geq -\frac{1}{2}x_2^2 \cdot sgn(x_2) \\ 1, & \text{otherwise} \end{cases} \tag{2.5}$$

When $x_1 < 0$

$$u = \begin{cases} 1, & x_1 \leq -\frac{1}{2}x_2^2 \cdot sgn(x_2) \\ -1, & \text{otherwise} \end{cases} \tag{2.6}$$

In Eqs. 2.5 and 2.6, *sgn* is the signum function. The analytical solution is plotted in Fig. II.2 for a case where the evader begins ahead of the pursuer with an initially larger velocity. As shown in the third subfigure, the minimum-time optimal control is bang-bang. It is assumed that both players have full and correct knowledge of the other's state.

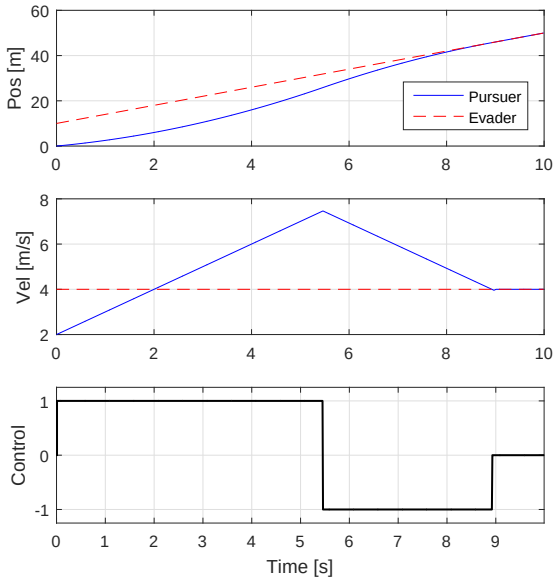


Figure II.2: Analytical feedback controller for 1D-rendezvous example

II.C. Receding Horizon Solution

A receding horizon approach to this problem cannot employ a minimum-time objective function J_p to find a solution, since the solution is found on repeated finite-time horizons. Instead, the objective J_p and optimal control sequence $\boldsymbol{u}^*$ can be rewritten in terms of the relative states along each finite trajectory. For a finite

planning horizon t_H, the cost is evaluated over the range $[t_0, t_0 + t_H]$ where t_0 is the time at the start of the current horizon. In Eqs. 2.7 and 2.8, h is the least-square function to minimize over, which is the relative state vector $[x_1 \ x_2]^T$. A terminal penalty m may be included, and weighting matrices Q and S may be used to penalize individual contributions more heavily.

$$J_p = \frac{1}{2} \sum_{t_0}^{t_0+t_H} \|h\|_Q^2 + \frac{1}{2}\|m\|_S^2 \tag{2.7}$$

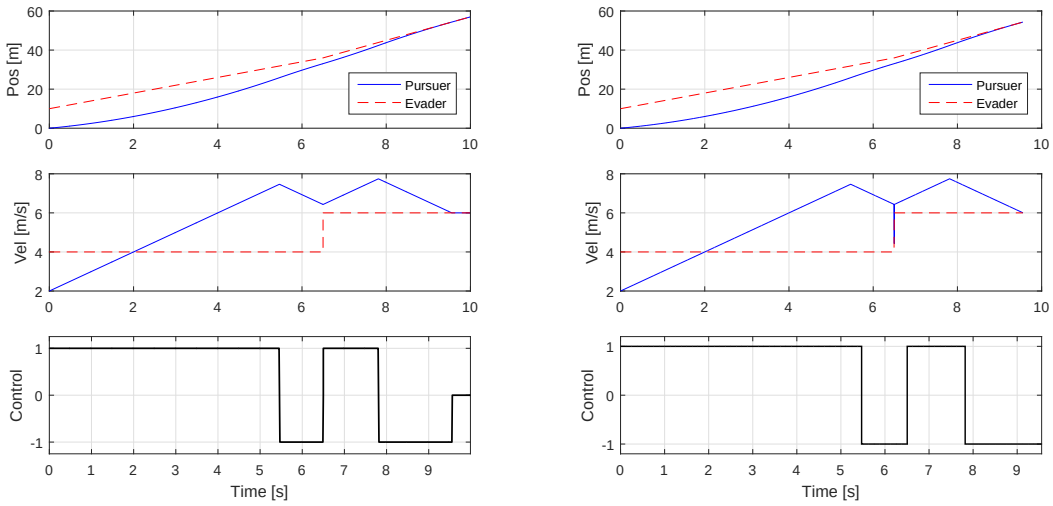
$$\begin{aligned} u^* &= argmin \frac{1}{2} \sum_{t_0}^{t_0+t_H} \|h\|_Q^2 + \frac{1}{2}\|m\|_S^2 \\ &= argmin(J_p) \end{aligned} \tag{2.8}$$

Of the optimal control sequence $\boldsymbol{u^*} = \{u_{t_0}^*, u_{t_1}^*, ...u_{t_0+t_H}^*\}$, only the first term $u_{t_0}^*$ is applied and held constant during the execution period t_E until the algorithm repeats. Further details of the RHC method including tuning and choice of planning and execution periods t_H and t_E are left for Chapter III.

II.D. Performance with Uncertainty

The behavior of a target may change at any time during a PE scenario, which is why it is dangerous and naive to assume a constant model and strategy. For example, an evader may be moving passively at the start of an encounter, then suddenly switch to an optimal course. For a one-dimensional game with unlimited fuel quantity and constant velocity, the evader's strategy and motion is fully determined. However, a simple way to simulate this dynamic uncertainty for performance comparison purposes is to allow the evader to instantaneously increase or decrease its speed within some range. As shown in Fig. II.3, both the analytical feedback and

RHC methods adjust control/trajectory as soon as the change in evader speed is sensed, since it is assumed that the velocity v_e is continuously and correctly sensed by the pursuer. A more comprehensive comparison of the controllers' performance is done by varying the amount that the speed changes Δv_e, as well as the time t_1 that it occurs. A batch of test cases was run by varying $\Delta v_e \in [-2, 2]$ m/s at increments of 0.1 m/s, at intermediate times $t_1 \in [4.0, 8.5]$ s at increments of 0.50 s. It can be seen in Figs. II.4 and II.5 that the RHC method performs equal to or slightly worse than the analytical solution, but without meaningful loss in optimality. The very few results that appear to lie below zero are due only to numerical tolerance. The worst instantaneous possible LO across the test cases is less than five percent, and the average across all cases is 1.57%.



(a) Analytical (b) RHC

Figure II.3: Performance with instantaneous evader speed jump at $t_1 = 6.5$ s; The solution is almost equivalent between methods

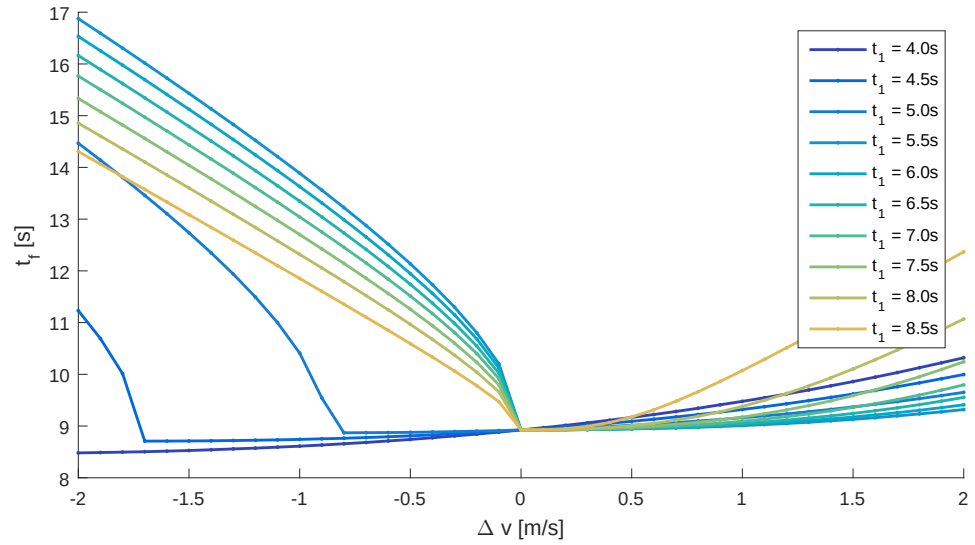


Figure II.4: Range of analytical test cases showing variation in capture time as a function of Δv_e and t_1

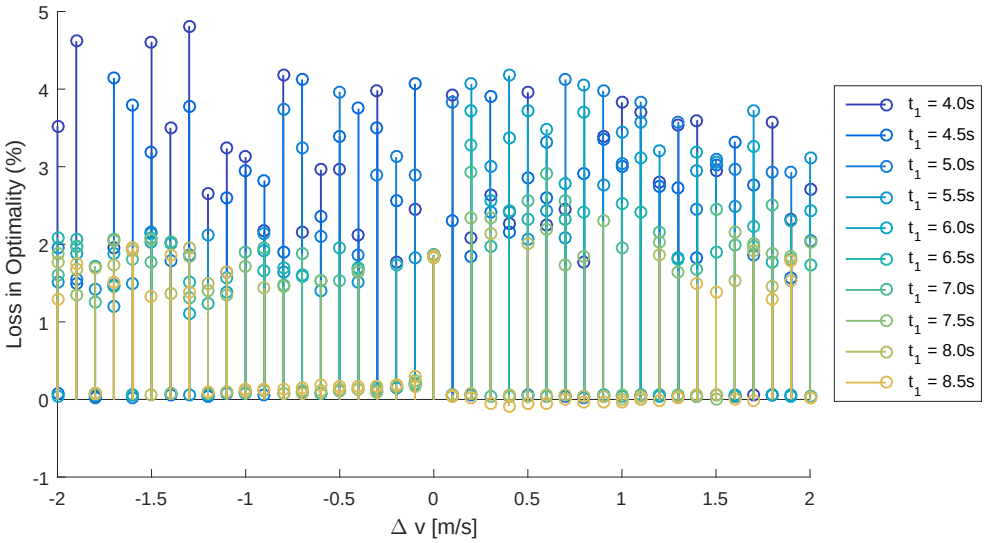


Figure II.5: Loss in optimality (LO) between analytic and RHC solutions as a function of Δv_e and t_1

II.E. Real-Time Considerations

MATLAB is an effective and straightforward way to write up scripts and simulate PE games, but the true test of these algorithms' practical usefulness can only be validated on hardware. Before compiling and testing on hardware, it is also important to have an idea of real-time performance. The MATLAB compiler and run-time environments cannot perform the computations for RHC coupled with fast system dynamics, and CPU times often exceed simulation times. This is shown in Fig. II.6. For the double integrator PE problem, a RHC solution that performs within 1% of the analytical requires an execution time of $t_E = 0.005$ s. At 200 Hz, the required CPU time per step is often longer than the simulated step size. This is particularly noticeable at the initialization of the game and minimization routine, and toward the end of the game where the minimization gradient decreases and optimization grows more taxing. On hardware, with added communication latency, this becomes a bigger problem as the computation and control commands will lag the sensed current states. A common design process is to move from MATLAB simulation to a real-time simulation, and then from the real-time simulation to hardware implementation. For real-time simulation, several open-source and commercial tools exist. The chosen tool for implementing the simulations in C++ is the ACADO toolkit, which includes a code generation tool for translation from simulation to executable run-time software [11, 12]. Using ACADO to solve the RHC with a cost function derived from Eq. 2.7 returns a result very close to the optimal infinite-horizon solution. Figure II.7 shows the RHC solution to the double integrator rendezvous problem. Note that the solution is almost identical to Fig. II.2, but that the bandwidth of the

controller in a real-time implementation (*i.e.* the effect of t_E) and the change in cost function J_p both affect the solution and $\boldsymbol{u}^*$ only closely approximates bang-bang. For this simple 1D example, ACADO offers a speedup over MATLAB routines and improved performance at slower sampling rates. With an average computation time of 27 ms[1], implementation using ACADO allows the algorithm to be run real-time in the loop.

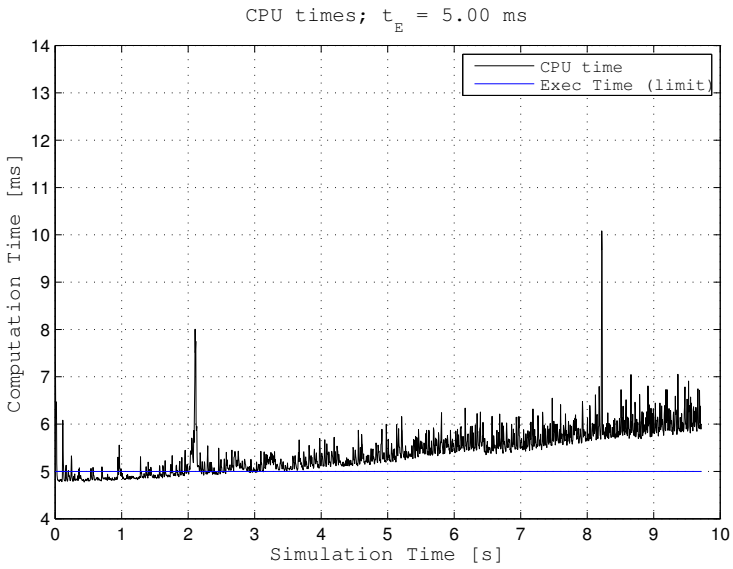


Figure II.6: Simulation time *vs.* computation time at each step; MATLAB simulation of the double integrator example

[1]Computation times given for a planning horizon $t_H = 2.0$ s and execution period $t_E = 0.10$ s

II.F. Summary

In this chapter, a one-dimensional example problem was shown in order to motivate the use of the RHC method. The loss in optimality of the RHC solution for this simple example is roughly 1 percent of the analytical solution capture time. Additionally, it appears that RHC handles strategic uncertainty and dynamic changes as well as a feedback solution if the execution period t_E is sufficiently short. Finally, the fast numerical solution of a repeated finite-horizon solution using NLP methods shows that the approach may be more viable for real-time operation on hardware than others that require repeated numerical minimization over an infinite horizon. These results promote the investigation of a numerically approximated RHC for more complex and uncertain systems. This is the subject of Chapter III.

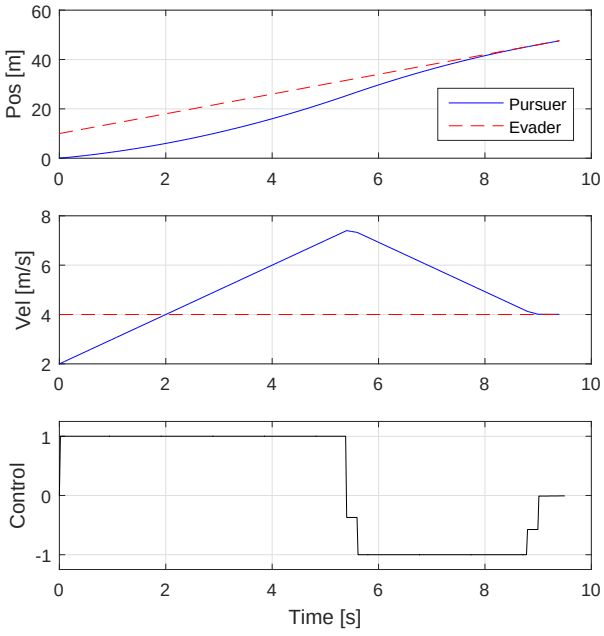


Figure II.7: RHC computed in real-time using ACADO solver

CHAPTER III

THE HOMICIDAL CHAUFFEUR

In Chapter II, properties and performance of a suboptimal RHC solution were examined for a simple one-dimensional example. Here, it is desired to extend the results and observations to higher-dimensional models with more interesting dynamics. The logical next step is to apply the RHC method to a two-dimensional planar system with nonlinear dynamics. It should be simple enough that the dynamics themselves are well understood and focus can be put toward control strategies and optimality. To this end, the homicidal chauffeur is used as the game model for the remainder of this work.

III.A. Problem Statement

The homicidal chauffeur is a PE game traditionally played with one pursuer and one evader. It is a model with several properties that make it illuminative for game-theoretic controls development. The system is nonlinear in both the pursuer and evader dynamics. The evader may instantaneously change its heading θ_e, while the pursuer is constrained by a dynamic equation and control constraint $-1 \leq u \leq 1$. The velocities and initial conditions can be chosen to ensure capture in finite time, so that the game has a solution. Commonly, the velocities are held constant to lend further insight to variation of other model parameters. The dynamics of the system can be written as a set of first order differential equations in either the inertial frame or the pursuer's body frame. The inertial representation is given in Eqs. (3.1) -

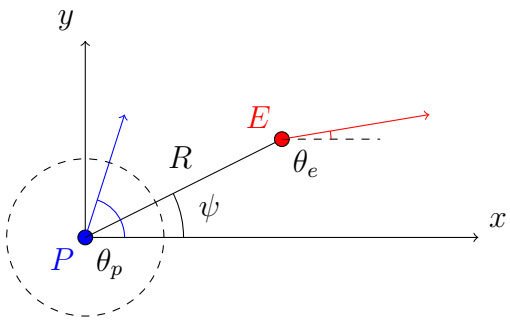


Figure III.1: Homicidal chauffeur reference frame

(3.5), and Eqs. (3.6) - (3.7) define the relative equation forms. Table III.1 defines and describes the referenced variables, and Fig. III.1 depicts a possible initial condition of the game.

$$\dot{x}_p = v_p \cos(\theta_p) \tag{3.1}$$

$$\dot{y}_p = v_p \sin(\theta_p) \tag{3.2}$$

$$\dot{\theta}_p = -\frac{v_p}{r_p} u \tag{3.3}$$

$$\dot{x}_e = v_e \cos(\theta_e) \tag{3.4}$$

$$\dot{y}_e = v_e \sin(\theta_e) \tag{3.5}$$

$$\dot{x}_1 = (-\frac{v_p}{r_p} u) x_2 + v_e \cos(\theta_e) \tag{3.6}$$

$$\dot{x}_2 = (\frac{v_p}{r_p} u) x_1 + v_e \sin(\theta_e) - v_p \tag{3.7}$$

where (x_1, x_2) are the evader coordinates and the system is in the pursuer's reference frame with the x_2 axis along its velocity direction.

Table III.1: Variables and default values for homicidal chauffeur simulation

Variable	**Value**	**Description**
v_p	1.0	Pursuer velocity; constant
v_e	0.4	Evader velocity; constant
r_p	0.5	Minimum pursuer turning radius
R_{cap}	0.8	Capture radius
x_{e_0}	2.0	Initial evader position
y_{e_0}	1.0	Initial evader position
R_0	$\sqrt{x_{e_0}^2 + y_{e_0}^2}$	Initial range
ψ_0	$\tan^{-1}(y_{e_0}/x_{e_0})$	Initial line of sight
θ_{p_0}	$\pi/2$	Initial pursuer heading

III.A.1. Assumptions

For the homicidal chauffeur simulations, several assumptions are made:

1. Velocities of each player are constant, and $v_e < v_p$
2. $R_0 > R_{cap}$
3. R_0 is sufficiently large such that $\epsilon_p \triangleq \frac{r_p}{R_0} \ll 1$
4. The game is played on an open plane with zero obstacles
5. Measurements are of range and bearing and are deterministic, such that all states can be extracted from each measurement

III.B. Feedback Solution

Unlike the 1D example in Chapter II, an analytic control solution to the homicidal chauffeur game does not exist. A solution must be found by numerically solving the OCP as a set of constrained nonlinear multivariable equations. This can be done using direct transcription and collocation with cubic Hermite interpolating polynomials according to the methods of Hargraves and Paris [13]. The problem is then solved using MATLAB's *fmincon* routine for constrained minimization [14, 15]. The result is an open-loop optimal control sequence. It is shown in the comprehensive

HC literature that for the minimum-time capture of this game, the optimal control sequence of the pursuer is either a bang-bang sequence with saturated controls or a bang-off sequence with a singular trajectory [16]. Further, for problem classes with sufficiently small ratios of r_p/R_0, the optimal control is of the second type. With a bang-off control sequence, the pursuer's trajectory is an initial turn toward its final heading followed by a straight segment.

However, direct transcription using *fmincon* can become computationally prohibitive and is sensitive to initial guesses. For problems where the initial pursuer heading may be clockwise or counter-clockwise, the time of capture can vary substantially. This makes it hard to accurately estimate capture time and populate the solver with good initial guesses. Instead, a feedback control that relies on the existence of a singularly perturbed trajectory (SPT) is used.

III.B.1. Singular Perturbation Technique

Singular perturbation theory is an energy-based method making use of a two-timescale phenomenon. When a system of differential equations can be split into two subsystems with a small parameter ϵ multiplying the second, it will be such that the second reaches some equilibrium state much faster than the first, and can be approximated as steady-state [17]. For example, consider Eqs. 3.8-3.9.

$$\dot{x}_1 = f_1(\boldsymbol{x}) + \epsilon g_1(\boldsymbol{x}) \tag{3.8}$$

$$\epsilon \dot{x}_2 = f_2(\boldsymbol{x}) + \epsilon g_2(\boldsymbol{x}) \tag{3.9}$$

If the perturbation parameter $\epsilon \ll 1$, they can be approximated by Eqs. 3.10-3.11.

$$\dot{x}_1 = f_1(\boldsymbol{x}) \tag{3.10}$$

$$f_2(\boldsymbol{x}) = 0 \tag{3.11}$$

SPT methods are applied to aerospace interception problems in [18] and [19]. A well-known use is the steady-state approximation of an aircraft phugoid mode using the time-scale separation of the short-period dynamics.

III.B.2. SPT Application to HC

In the HC game model, the assumptions include that the pursuer's minimal turning radius r_p is much smaller than the initial range between players R_0. In this scenario the angular velocity of the pursuer evolves on a faster timescale than the inter-vehicle range. The perturbation parameter for SPT here is $\epsilon_p \triangleq r_p/R_0$. Since the optimal control for small ϵ_p is an initial turn toward the final line of sight (LOS), the solution is known as soon as ψ_f is known. Using the reduced order model as in [18], the final line of sight (LOS) ψ_f between the two players can be approximated by $\bar{\psi}_f$ using a geometric construction. The approximate optimal control becomes a function of $\bar{\psi}_f$ and does not require numerical minimization or an initial guess of capture time. Note that when the current LOS ψ equals the evader bearing, $(\psi - \theta_e) = 0$ and Eq. 3.13 is negated and the game-optimal trajectories become a straight "chase", *i.e.* $\theta_p = \psi_f = \theta_e$.

$$u^* = sign(\theta_p - \psi_f) \tag{3.12}$$

$$\Delta\psi \cong \tan^{-1} \frac{(v_e/v_p)sin(\psi - \theta_e)}{(R/R_{cap}) - (v_e/v_p)\cos(\psi - \theta_e)} \tag{3.13}$$

$$\bar{\psi}_f = \psi + \Delta\psi - \sin^{-1}[(v_e/v_p)\sin(\psi + \Delta\psi)] \tag{3.14}$$

$$u = sign(\theta_p - \bar{\psi}_f) \tag{3.15}$$

The SPT feedback solution is compared to solutions found using both the MATLAB direct transcription routine *fmincon* and the nonlinear equality solver *fsolve*. Both MATLAB routines determine local minima only, but with well-informed initial

guesses about the solution should converge to the global minimum. Table III.2 shows the variation in capture time according to each method. The direct OCP solution yields the lowest capture time for each case. The average LO of the feedback approach using $\bar{\psi}_f$ is 0.117%. Additionally, the feedback approach performs on average 0.438% better than the equality solver *fsolve*. Due to its performance quality, faster computation, and insensitivity to poor initial guesses, the feedback control method using SPT analysis is implemented henceforth.

Table III.2: Capture times for homicidal chauffeur using various solution methods; Times are in seconds

v_e/v_p	Evader Heading θ_e	t_f (*fmincon*)	t_f (feedback with $\bar{\psi}_f$)	t_f (*fsolve*)
0.10	0	1.7296	1.730	1.7443
0.10	$\pi/4$	1.7181	1.720	1.7521
0.10	$-\pi/6$	1.6816	1.685	1.6938
0.40	0	2.5565	2.560	2.5576
0.40	$\pi/4$	2.4861	2.490	2.4873
0.40	$-\pi/6$	2.2734	2.275	2.2734

III.B.3. Uncertain Player Behavior

The evader may not be playing as expected. An additional decision the pursuer may have to make is what strategy to play, based on an estimated strategy of the evader. There are several ways of doing this. Behavior learning techniques can be used to estimate and learn the strategy of an opponent, then transform the PE game into a one-sided optimal control problem [8]. This framework has been shown to be successful for several applications when combined with linearized system models [8, 9]. Unfortunately, behavior learning adds a layer of complexity to the player's computations and, like nonlinear numerical minimization problems, has not yet been

demonstrated on simple hardware.

An approach to handling uncertainty without behavior learning augmentation is to make a simple guess of the other player's strategy. The differential zero-sum solution does this by assuming both players are playing their best strategy. A surface map of capture times can be generated according to independent control variables. If both players are playing optimally, some ceiling for the time of capture can be established as the surface's saddle point. This is illustrated on the simple saddle surface $z = x^2 - y^2$ shown in Fig. III.2. If the evader plays anything other than its optimal strategy, the pursuer gains performance. This is a robust and effective way to handle strategic uncertainty.

A second guess one could make is that of a passive evader. Here, "passive evader" means the evader enacts a zero order hold (ZOH) on its control until the next sample instant. During a ZOH in the homicidal chauffeur game, the dynamics of the evader continue evolving but there is no directional or speed change permitted. In this sense, the evader is behaving passively with regard to its chaser. Regardless of which strategy is assumed, an incorrect guess decreases performance and possibly (in some cases) results in capture failure.

To this end, four extremal cases were examined. In two cases the evader enacts a ZOH ($u_e = 0$) and does not alter or optimize its heading θ_e. In the other two cases it plays the game-optimal solution ($u_e = u_e^*$) and the optimal heading θ_e^* is chosen. For each evader strategy, the pursuer either plays the traditional zero-sum game strategy, assuming $u_e = u_e^*$, or an optimal control strategy assuming that $u_e = 0$. In the latter, the problem ceases to be a typical PE game and is better described as a single OCP against a moving target with known dynamics. The optimal evader heading θ_e^* is determined a priori as the heading $\theta_e \in [0, 2\pi]$ that maximizes the capture time t_f of the pursuer. For the initial conditions in Table III.1, $\theta_e = \theta_e^* = 0.33$ rad. Table

III.3 shows that there is less potential performance loss associated with the pursuer playing the optimal strategy, as opposed to the pursuer playing the game-optimal zero-sum strategy. This matches previous literature results for a similar SPT problem class [5].

Table III.3: Capture times for homicidal chauffeur with various pursuer and evader strategies; $(v_e/v_p) = 0.40$

	$u_p = \bar{u_p}$	$u_p = u_p^*$	LO %
$u_e = 0$	2.557 s	2.566 s	0.352%
$u_e = u_e^*$	2.616 s	2.607 s	0.269%

To confirm this somewhat unintuitive result, a closer look at the zero-sum solution is required. Consider the saddle point solution to a zero-sum game based on the homicidal chauffeur model. One could model the saddle surface as a function of two variables, $t_f = f(\theta_e, t_s)$. Here, the downward-inflection of the surface is mapped by constant contours of evader heading θ_e. The optimal θ_e^* will be the highest point on any contour. The upward-inflection of the surface is mapped by constant contours of what can be referred to as the switching time t_s. This is the time at which the pursuer switches its control authority from a saturated point (± 1) to zero. It has already been shown that the bang-off control law is optimal for the HC problem class with the given assumptions [16]. The optimal switch time for a deterministic zero-sum game t_s^* will minimize the upward-inflection surface of the saddle and guarantee a capture time ceiling.

At this point one still is not considering any specific game scenarios, but simply forming the solution surface and observing its geometry. If the evader and pursuer both play optimally, the solution point t_f will lie at the stationary point, or saddle

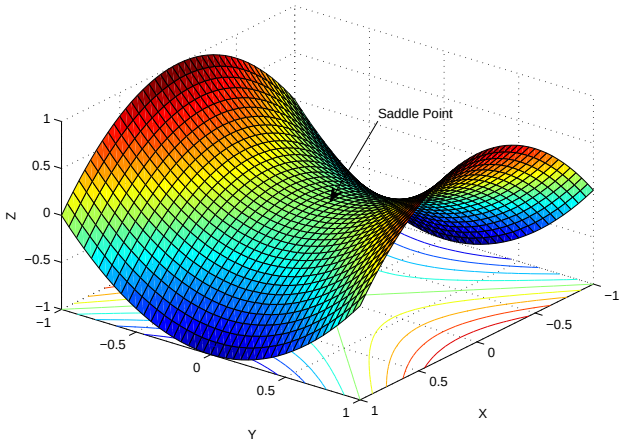


Figure III.2: A simple saddle surface model

point. If the pursuer or evader deviate from their best control, the solution will deviate from the saddle point. So, one should naturally believe that if it can be proven that the zero-sum strategy repeated in Eq. 3.16 yields a different t_s than the optimal strategy in Eq. 3.17 for the same problem, one strategy could indeed outperform the other depending on θ_e and t_s.

$$u^{zs} = \min_{u_p} \max_{u_e} \left(\int_{t_0}^{t_f} dt \right) \tag{3.16}$$

$$u^{*} = \min_{u_p} \left(\int_{t_0}^{t_f} dt \right) \tag{3.17}$$

The actual contour surface for $t_f = t_f(\theta_e, t_s)$ is shown in Fig. III.3. Only a subset of the whole surface is shown: $\theta_e \in [-0.4, 0.4]$ rad captures both passive and optimal evader bearings for the initial conditions given earlier. t_s is discretized on $t_s \in [0.5, 1.0]$ seconds. Figure III.4 is a finer grid of the area showing extremal results that agree with Table III.3. Discretization across an even finer grid could be done but it is computationally expensive to do so for purely illustrative purposes, hence the t_f values differ quantitatively only. The results of this section also highlight one other important note about game-theoretic approaches to controls: it is very unwise

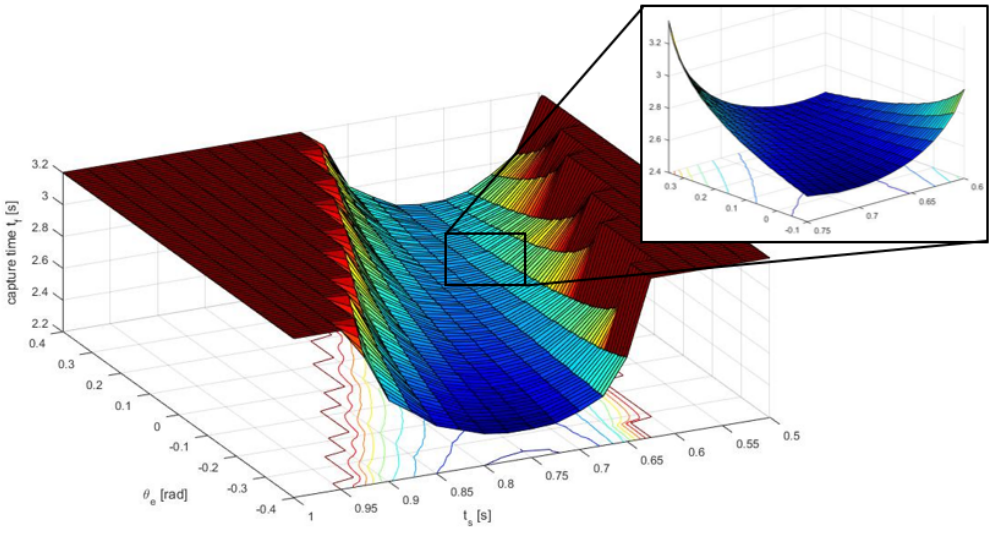


Figure III.3: Contour surface for homicidal chauffeur problem

to play a game scenario with an open-loop control. Although capture still occurs for each case shared in Table III.3, an open-loop control based on t_s would usually fail to yield capture if the evader changed its behavior partway through the encounter.

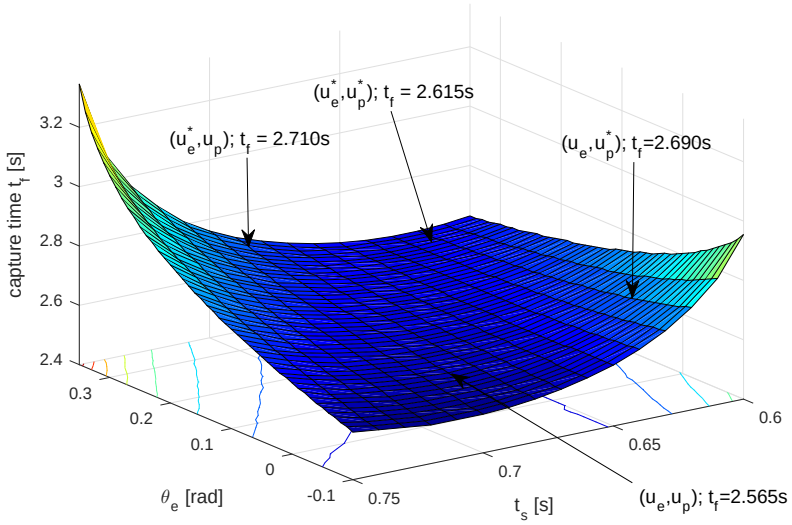


Figure III.4: Contour surface (close-up) for homicidal chauffeur; Labeled extremal solutions

III.C. Receding Horizon Control

It has been shown that the evader ZOH assumption is a superior approach when evader behavior is uncertain. Regardless of the strategy chosen, augmentation with a receding horizon controller offers additional benefits. Full determination of the infinite horizon solution at each sample point is too computationally heavy to do in the loop, but a single open-loop solution or perturbation control about a nominal trajectory is also a bad idea if the reference trajectory is uncertain or changes partway through the game. A RHC scheme with short execution times t_E to correct for erroneous information further improves the performance of a controller operating on the evader ZOH assumption.

The optimal finite-horizon control can be found during each planning horizon by minimizing against a set of relative states like in Chapter II. For minimum-time capture or rendezvous, the minimization of relative states (with no control penalty) is aligned with minimal time. However, it is known that a SPT approximation to the

infinite-time problem exists and yields extremely accurate results. Even better, it does not require an accurate (or any!) initial guess. If a similar SPT approximation can be made for the finite-time problem, one could determine $\bar{\psi}(T)$ and express u directly without numerical minimization. Eq. 3.18 does just that; the approximation for ψ is made for the known final time $T = t_0 + t_H$. Subscripts 0 indicate the state or time at the start of the current planning horizon.

$$\bar{\psi}(T) = \bar{\psi}(t_0 + t_H) \cong \tan^{-1}\left(\frac{R_0 \sin\psi_0 + v_e(T - t_0)\sin\theta_{e_0}}{R_0 \cos\psi_0 + v_e(T - t_0)\cos\theta_{e_0}}\right) \tag{3.18}$$

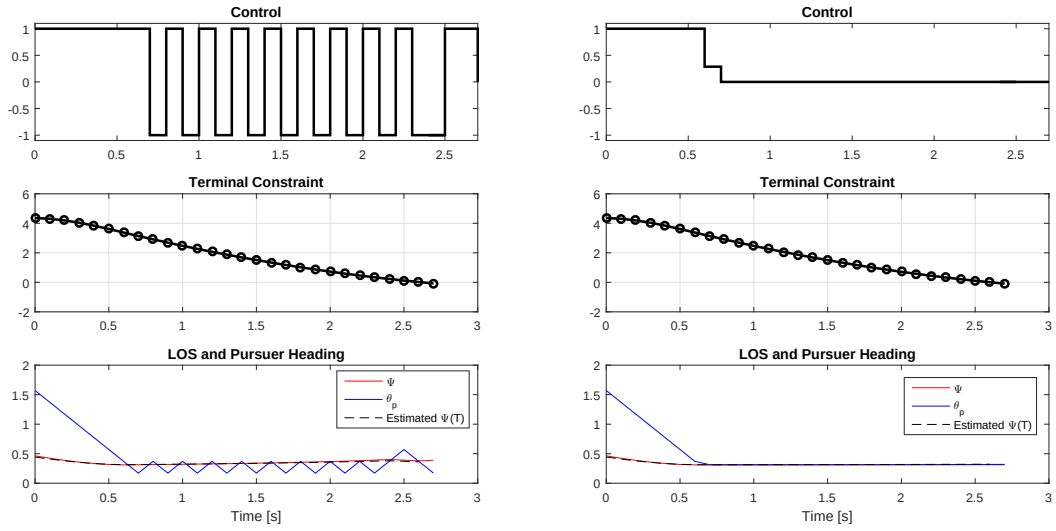
$$u = sign(\theta_p - \bar{\psi}(T)) \tag{3.19}$$

III.C.1. Controller Bandwidth

The RHC-SPT approximation of $\bar{\psi}(T)$ allows us to bypass typical cost function minimization and significantly reduce on-board computation. Eqs. 3.15 and 3.19 indicate that the approximate optimal control is bang-bang or bang-off. However, controller bandwidth considerations lead one to modify the implemented controller slightly. At infinite bandwidth ($t_E \longrightarrow 0$), the controller can easily reach $\theta_p = \bar{\psi}_f$ or $\bar{\psi}(T)$. Yet, at a realistically limited bandwidth of 10-100 Hz, the heading θ_p will almost always pass the desired LOS heading ψ between control instances. This results in an oscillatory behavior around $\bar{\psi}(T)$, as shown in Fig. III.5a. To eliminate this undesirable behavior, a controller modification is made. A tolerance ξ is first defined to introduce a control deadband. Then, the bang-off control law is further modified by adding a scaled control term that is applied whenever the heading error falls between ξ and the maximum heading change per sample. The resulting control structure given in Eq. 3.20 is applied in simulation and Fig. III.5b shows the improvement in heading convergence and the decrease in control effort. While there is little performance gain realized in simulation with this modification, the trajectory

is smoothed and the significant decrease in control effort would be beneficial in hardware.

$$u = \begin{cases} 0, & |\ (\theta_p - \bar{\psi}(T))\ | \leq \xi \\ (\theta_p - \bar{\psi}(T))\frac{r_p}{v_p t_E}, & \xi < |\ (\theta_p - \bar{\psi}(T))\ | < \frac{v_p}{r_p} t_E \\ sign(\theta_p - \bar{\psi}(T)), & |\ (\theta_p - \bar{\psi}(T))\ | \geq \frac{v_p}{r_p} t_E \end{cases} \tag{3.20}$$



(a) Original bang-off

(b) Modified bang-off; $\xi = 0.02$

Figure III.5: Comparison of the original bang-off RHC controller using sign-based control only and the modified controller given in Eq. 3.20; Sampled at 10 Hz

III.C.2. Controller Stability

Stability analysis is an important part of controller design, especially for nonlinear systems where the output behavior is not easily predicted for arbitrary system configurations and initial conditions. An extensive investigation of the stability and robustness of various nonlinear receding horizon controllers exists in the liter-

ature. In [20] and [21], the stability of unconstrained nonlinear RHC controllers with quadratic cost functions are examined. [22] investigates constrained RHC control and its robustness. [23] considers the stabilization and robustness of RHC and dual-mode RHC controllers with terminal constraint regions, and [24] proves the stability of a sampled-data RHC controller that is derived from a stable continuous RHC controller. In [25] it is shown that when terminal constraints are coupled with short planning horizons, systems may lose any guarantee of robustness. Finally, [26] demonstrates examples where naive sampled-data RHC control may actually destabilize a model that is stable under continuous-time control. The theme of these papers is the potential destabilizing nature of a RHC cost function without terminal constraints, and its lack of robustness when coupled with naive planning horizon and control bandwidth selection.

The homicidal chauffeur control algorithm is a nonlinear receding horizon, sampled-data controller with control constraints. However, each of the cited works make assumptions about the existing stability or conditions of the system and/or controller, and the results are not directly applicable to the problem at hand. Instead, stability and robustness is evaluated numerically and used to show a practical robustness of the controller.

The approach taken to show practical stability and robustness of the controlled system is to perform several large Monte Carlo analyses on the homicidal game simulation with the SPT-RHC controller. Practical asymptotic stability is considered as successful capture within a time $t \in (0, t_{limit}]$ where t_{limit} may be set to a large upper expected value of capture, given the distribution of possible initial conditions. Practical robustness is considered as practical asymptotic stability under appropriately characterized Gaussian disturbances $d \sim N(0, \sigma_d)$.

Several Monte Carlo analyses show with 100% confidence that the algorithm

returns a robust asymptotically stabilizing controller under assumptions A-1 to A-3. The Monte Carlo simulation parameters and results are presented in the Appendix.

Assumption A-1. *The planning horizon t_H may be arbitrarily small > 0 provided that it is greater than or equal to the length of the execution time t_E. That is,* $0 < t_E \leq t_H$.

Assumption A-2. *The initial state of the evader $(x_e(0), y_e(0)) \in \mathcal{S}$, where $\mathcal{S}$ is defined as the open set bounded only by the pursuer's initial line of sight, LOS $\psi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$.*

Assumption A-3. *The control period t_E is 'short enough' to reasonably control the pursuer vehicle without consideration of the PE game.*

What is meant by A-3 is that very slow controls at $t_E = \{2, 5, 10...\}$ s do not result in successful capture and game completion within t_{limit}, but they are not reasonable or practical control rates for the pursuer vehicle to begin with. For the homicidal chauffeur time-scale, a control rate of at least 2 Hz is reasonable, and slower controllers should be considered with caution.

III.C.3. Numerical Controller Comparison

To this point, three methods for finding a control law to achieve minimum-time capture have been presented. The SPT "feedback" algorithm which recursively estimates the infinite-horizon line of sight $\bar{\psi}_f$ is shown to be successful to within 1% of the optimal solution and computationally superior. Additionally, the receding SPT algorithm is presented as a finite-horizon alternative that recursively and quickly estimates $\psi(\bar{T})$. To compare the two SPT methods, a numerical study is performed.

The aim of the study is to investigate a wider range of initial conditions and evader behavior and to confirm that the RHC-SPT solution does not degrade significantly under non-ideal conditions. Evader speed jumps Δv_e between -0.20 and +0.40 m/s at 0.025 m/s increments are applied at four different times t_1, each for three different evasive strategies. The first strategy is that of a purely optimal evader that does not change its heading θ_e^* during the simulation. The second changes from a passive ZOH at $\theta_e = 0\,\text{deg}$ to the optimal heading θ_e^* at t_1. The third begins at an optimal heading θ_e^* but changes heading to $\theta_e = -45\,\text{deg}$ at t_1. Figures III.6 and III.7 show the resulting capture times under each condition and the associated LO.

Figure III.7 illuminates several important differences between the SPT feedback and RHC-SPT approaches. When $\Delta v_e \in [-0.1, +0.1]$, the first approach performs 4-5% better. As the magnitude of Δv_e increases and the maneuvers become more dramatic, the RHC-SPT approach starts to outperform the previous. In these cases, the difference between the current time and the end of the horizon (t, T) is smaller than between the current time and estimated capture time $(t, \bar{t_f})$ and the faster corrective nature of the RHC-SPT algorithm is more effective at adjusting to the maneuvers. In the more drastic sample cases, the RHC-SPT algorithm significantly outperforms the SPT feedback method. An example is illustrative: for an optimal to suboptimal evasive maneuver with a $\Delta v_e = +0.40$ m/s and maneuver time $t_1 = 0.50$ s, the optimal control law yields $t_f = 3.675$ s. The RHC-SPT algorithm yields $t_f = 3.750$ s with only 2% LO, while the SPT feedback law results in capture at $t_f = 4.935$ s and 34% LO. Clearly, the RHC method pays dividends during quickly evolving and unpredictable game scenarios.

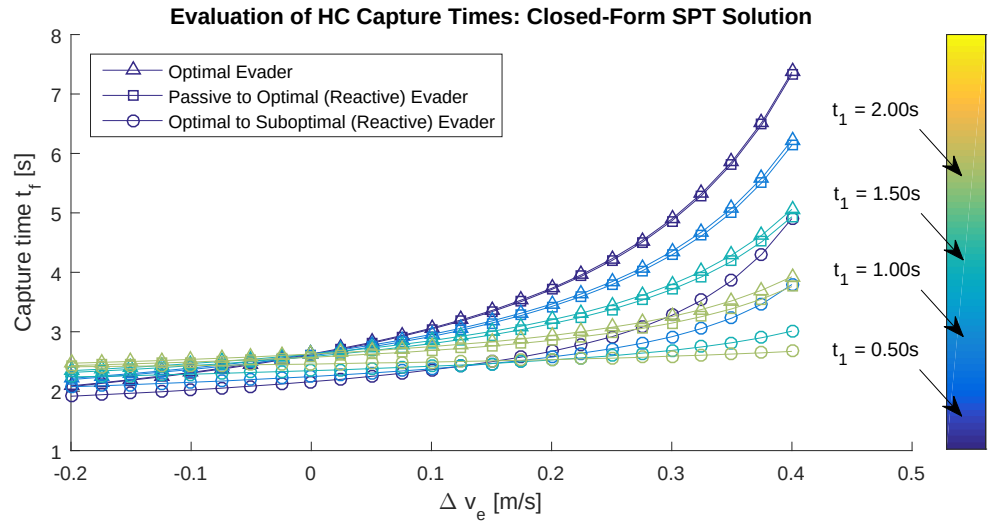


Figure III.6: SPT feedback numerical study; capture times

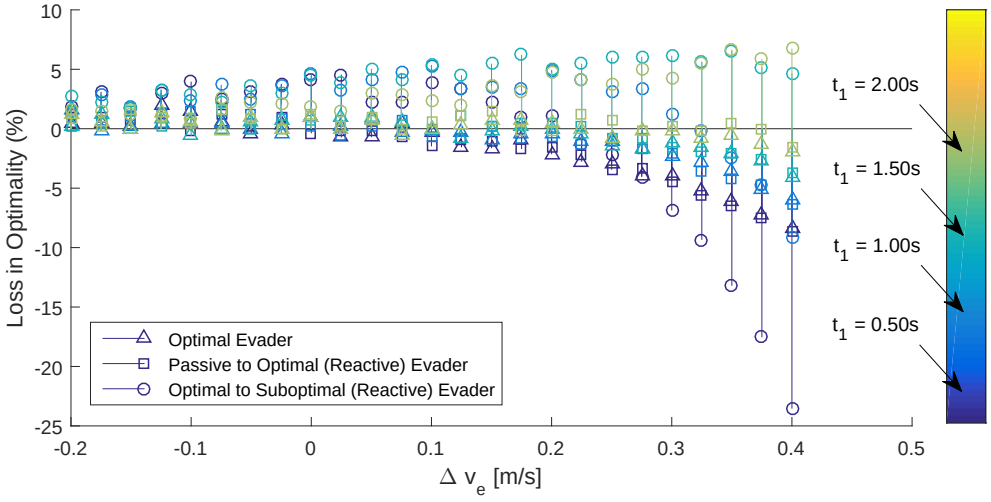


Figure III.7: Loss in performance between SPT feedback and RHC-SPT solutions

III.D. Simulation Results

Finally, some simulation results are presented. The full simulation of the uncertain homicidal chauffeur PE game with two players allows for any instantaneous evasive maneuver to be applied by the user. The results given here are for the RHC-SPT controller and were found using a planning horizon $t_H = 2$ seconds and an execution time or control rate of $t_E = 0.05$ seconds (20 Hz). Theoretically, a RHC law will always perform better as t_H is increased[1], as the longer finite-horizon length better approximates the infinite-horizon solution. With the SPT solution it can be seen that since $\bar{\psi}(T)$ is calculated using the states at the start of the horizon, some long horizons will actually start to degrade the approximation. Through trial and error, $t_H = 1 - 2$ seconds was found to best match the optimal capture time. All sampled-data controllers should perform best at high bandwidth, so a small t_E is ideal. Practically, t_E is limited by computing power and how much CPU one is willing to spend integrating each simulation run.

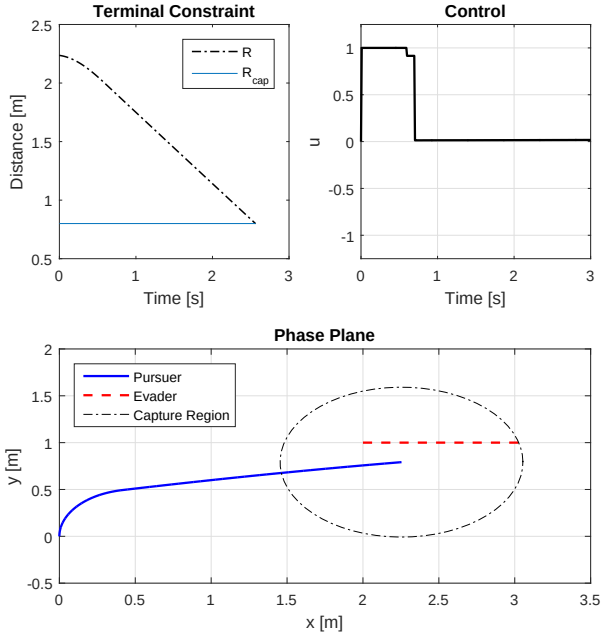
The simulation results shown in Figs. III.8 and III.9 were done initially in MATLAB, then repeated using the ACADO toolkit in C++ to incorporate the real-time computation. Plots show the terminal constraint in the upper left subfigure. This measures the inter-vehicle distance and the terminal condition $R \leq R_{cap}$. The control $\boldsymbol{u}$ is shown in the upper right subfigure to transition between saturated controls and singular arcs. Finally, each has a lower subplot showing the phase plane, or aerial view, of the two vehicles' chase path and the pursuer's capture radius.

In Fig. III.8, evader dynamics are constant and either passive or optimal. In Fig. III.9, the evader applies an instantaneous Δv_e and $\Delta\theta_e$ at some intermediate time. These changes represent a switch to or from the optimal heading at arbitrary times.

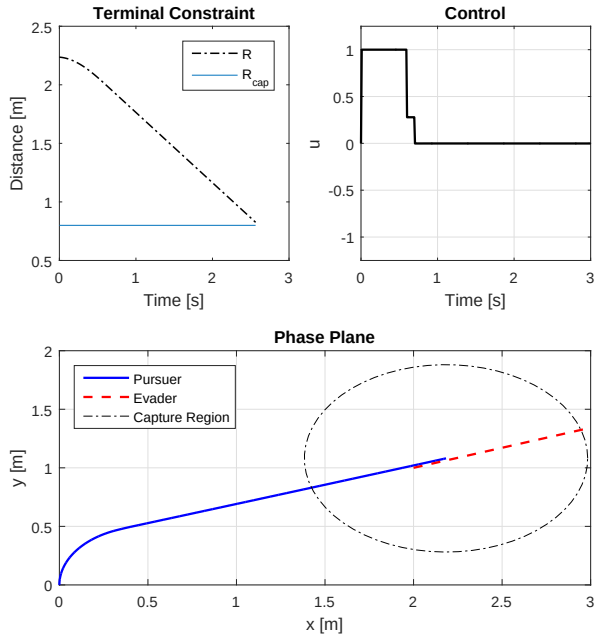
[1]Assuming the evader behavior is known

In each case, successful capture in minimum time is demonstrated. Any change to evader parameters is a deterministic measurement made by the pursuer at the start of its next planning horizon.

The results of these simulations strongly support the performance benefits of the developed RHC law and the validity of the SPT approach. The next step toward real-time validation is implementation and experimentation on hardware. This progression extends the results of [27] and is the subject of Chapter IV.

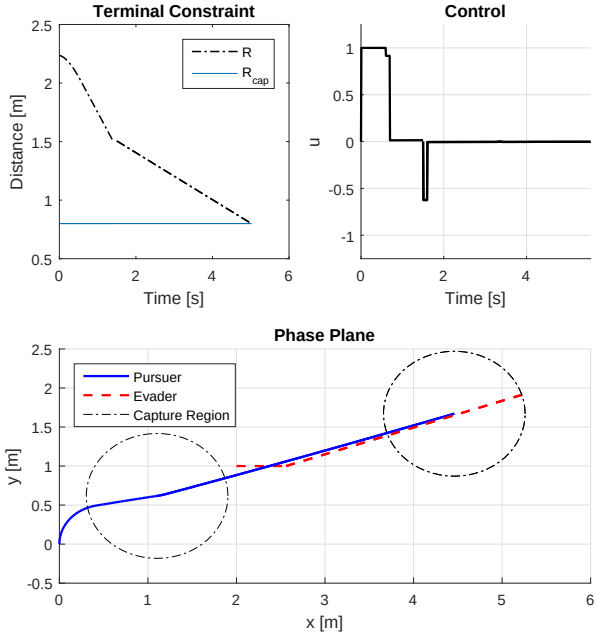


(a) Passive evader

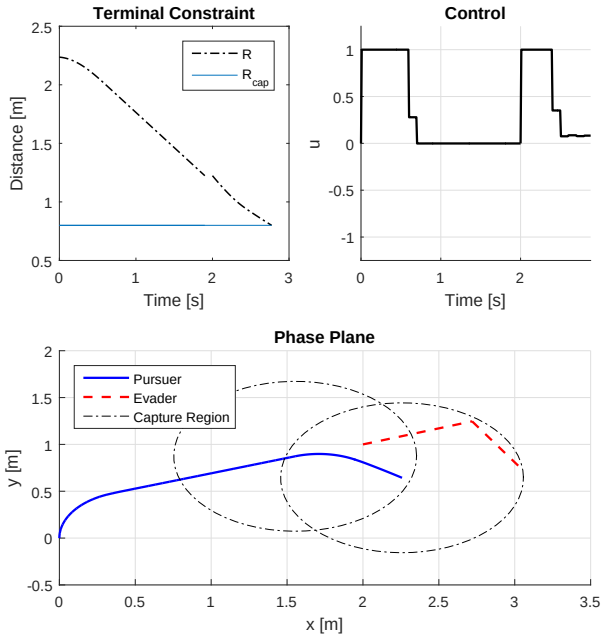


(b) Optimal evader

Figure III.8: Real-time receding horizon control results; Constant evader dynamics



(a) Change speed and heading at $t_1 = 1.5$ s



(b) Change heading at $t_1 = 2.0$ s

Figure III.9: Real-time receding horizon control results; Varying evader dynamics

CHAPTER IV
HARDWARE IMPLEMENTATION

IV.A. Motivation

Much can be learned from the work done in Chapter III. Simulated results show the efficacy of the receding horizon approach for uncertain pursuit-evasion games, especially for those with singularly perturbed trajectories. Further, simulation of the real-time algorithm in C++ demonstrates a drastic speedup compared to numerical simulation in MATLAB, and presents a viable way to compute receding horizon control laws online during actual scenarios.

However, to truly validate the approach for hardware use, there can be no better method than testing the code on actual hardware. This introduces physical realizations and limitations, integrates the computation with lower-level motor controls, and communicates with internal (IMU, Gyro, etc.) or external (motion capture) data for localization. Any of these factors may introduce real system disturbances, measurement noise, and communication latency. A successful experiment on test-bed hardware that continues to successfully approximate the optimal solution is stronger evidence of the method's effectiveness. To do the hardware validation, a laboratory test space is required.

The Land, Air, and Space Robotics laboratory at Texas A&M is well and uniquely suited to many of the guidance, control, and navigation problems that are currently relevant to aircraft and space systems research. NASA defines nine technology readiness levels (TRL) to measure the maturity level of a particular technology, as shown in Fig. IV.1. At TRL 1, only basic principles of the phenomena have been observed and reported. At TRL 9, an actual system has been field proven

through successful mission operations [28]. University and student research often lies within TRL 1-2 and sometimes 3. At Texas A&M, several laboratories are uniquely suited to progress toward TRL 4: component validation in a laboratory environment. LASR Lab is one such space and several project partnerships with industry have done component validation in a simulated space environment using actual sensors or hardware.

The work in this thesis is an application of existing theory, extended by new real-time implementation and testing of the methods using a classical problem. Therefore this characteristic proof-of-concept testing falls under TRL 3, but the RHC approach and uncertainty treatment could be applied to other problems on higher-fidelity vehicles and sensors at LASR lab in the future to continue satisfying NASA's path to technological readiness.

The LASR lab test space is shown in Fig. IV.2.

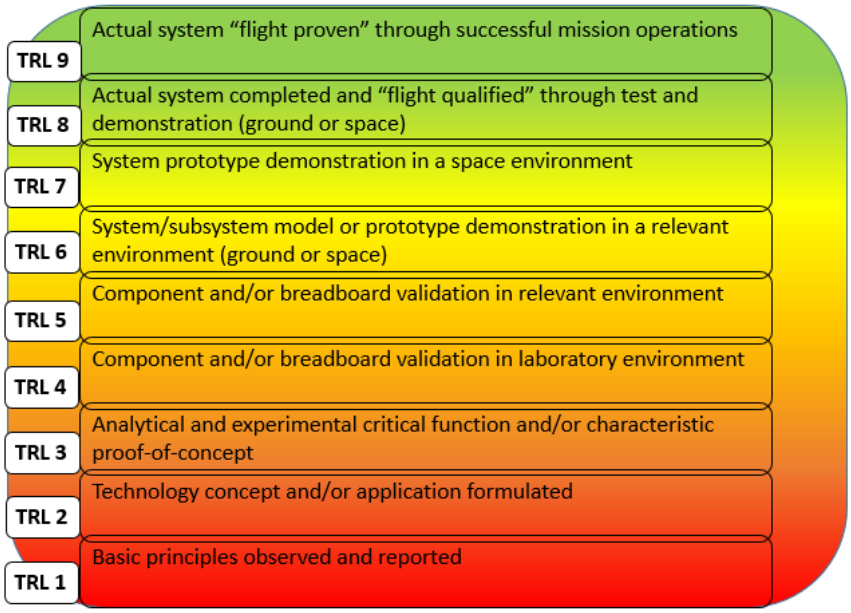


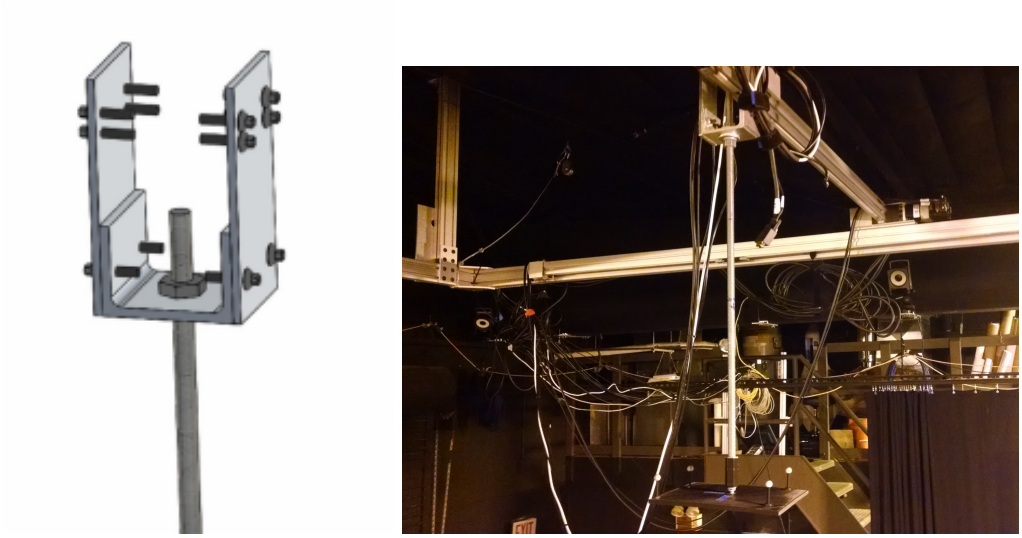
Figure IV.1: NASA Technology Readiness Level definitions

Figure IV.2: LASR Lab experiment space

IV.B. Vehicles

For HWIL testing, the pursuer vehicle is an iRobot® Create ground vehicle. The robot is a commercial hobbyist platform and is essentially a programmable ground robot with odometry, bump and cliff sensors, and serial communication port. The Create has a maximum speed of 0.50 m/s and a diameter of roughly 0.30 m. It can turn in place, but by artificially limiting its turn radius, one can use it to model any simple differentially-driven forward flight vehicle. Of course, it is a wheeled ground vehicle so its dynamics and the motion that it can emulate must be planar.

The Suspended Target Emulation Pendulum (STEP) is a custom-built platform at the LASR Lab that simulates free "space-like" motion in 5 degrees of freedom. The pendulum has a free-response mode that allows for contact dynamic and proximity operation experiments. The STEP also has a velocity-tracking mode, which allows it to simply follow commanded motor speeds. By specifying individual motor speeds along each axis, trajectories to simulate a translating body can be created. By using this velocity-tracking mode and affixing a small rigid body to the pendulum tip, evader motion with instantaneous velocity changes can be replicated; this makes use



(a) (b)

Figure IV.3: Rigid suspension assembly for hardware validation using the STEP

of the omni-directional motion capability of the pendulum.

However, the standard STEP system utilizes a long pendulum arm constructed of thin, flexible carbon fiber rod. When subjected to the dynamics of a quickly-turning evader, the swinging of the pendulum arm and the low-frequency oscillations of the rod result in oscillatory velocity at the STEP tip. To prevent this, a new rigid assembly was constructed. Since the crane along the top of the STEP correctly tracks with constant velocity and instantaneous directional capability, a new assembly was attached to it at the old hinge location. The assembly model is depicted in Fig. IV.3a and a close-up image of its attachment to the STEP crane is shown in Fig. IV.3b. The rigid assembly is comprised of aluminum bar and channel and a suspended 1/2" threaded steel rod. The thick rod has a larger modulus than carbon fiber, has high-frequency modes that are minimally excited by the low speeds of the tests, and is not hinged. As a result, the evader body can thread into the steel rod and very closely mimic the crane's behavior while inside the experiment space.

IV.C. System Communications

LASR Lab is equipped with several motion capture (MoCap) systems for sensing and reporting truth data both in real-time during experiments and for post-processing. One system is the ©Vicon motion tracking system. Vicon is extensively used in the entertainment and video recording industry to capture and record motion of human and other subjects. The MoCap system provides 1-millimeter resolution at up to 120 Hz using 16 megapixel infrared cameras. In the LASR laboratory space, six Vicon cameras are mounted above and outside the test area to get full coverage of experiments. To render an object trackable by Vicon, small retro-reflective beacons are mounted on the object. These markers are used to define a rigid body within the Vicon software and are then tracked together once the body is registered and recognized. This provides location and orientation tracking capability to the MoCap system.

The Vicon system measures seven states in real time: three translational states $[x, y, z]^T$ and the quaternion $[\beta_0, \beta_1, \beta_2, \beta_3]^T$. If one assumes that each vehicle is localized in its own frame and has sensors to measure the other vehicle's position and orientation, then these measurements represent what is realistically sensed. However, additional parameters of the evader motion, used in computation of the pursuit control law, may be required. In the RHC algorithm the evader speed v_e is required. This is treated as a known variable in simulation, but must be estimated in a laboratory frame. To do this, a simple tracking filter is implemented that filters real-time Vicon position data and predicts velocity components $[v_{e_x}, v_{e_y}, v_{e_z}]$. The filter is a *g-h-k* filter (equivalently: α-β-γ or zero-jerk filter) [29]. Under the assumption of zero jerk, the discrete equations of motion of the tracked target are given in Eqs. 4.1

- 4.3. The update and prediction steps are given in Eqs. 4.4 - 4.10.

$$x_{k+1} = x_k + \dot{x}_k \Delta t + \ddot{x}_k \frac{\Delta t^2}{2} \tag{4.1}$$

$$\dot{x}_{k+1} = \dot{x}_k + \ddot{x}_k \Delta t \tag{4.2}$$

$$\ddot{x}_{k+1} = \ddot{x}_k = \ddot{x}_0 \tag{4.3}$$

$$\bar{x}_{k,k} = \bar{x}_{k,k-1} + g(y_k - \bar{x}_{k,k-1}) \tag{4.4}$$

$$\dot{\bar{x}}_{k,k} = \dot{\bar{x}}_{k,k-1} + \frac{h}{\Delta t}(y_k - \bar{x}_{k,k-1}) \tag{4.5}$$

$$\ddot{\bar{x}}_{k,k} = \ddot{\bar{x}}_{k,k-1} + \frac{2k}{\Delta t^2}(y_k - \bar{x}_{k,k-1}) \tag{4.6}$$

$$\tag{4.7}$$

$$\bar{x}_{k+1,k} = \bar{x}_{k,k} + \dot{\bar{x}}_{k,k} \Delta t + \frac{1}{2} \ddot{\bar{x}}_{k,k} \Delta t^2 \tag{4.8}$$

$$\dot{\bar{x}}_{k+1,k} = \dot{\bar{x}}_{k,k} + \ddot{\bar{x}}_{k,k} \Delta t \tag{4.9}$$

$$\ddot{\bar{x}}_{k+1,k} = \dot{\bar{x}}_{k,k} \tag{4.10}$$

For models with non-zero jerk, a small bias exists. The *g-h-k* gains are determined to minimize the discounted least-square error for a constant-acceleration target given in Eq. 4.11, where y is the measurement and $p^*(r)$ is a fitted polynomial. This performance index may be referred to as a "fading memory" filtering scheme. A tuning parameter $\theta \in [0, 1)$ determines the weight of receding measurements and the update gains. A large θ weights past measurements more heavily than a small θ. For testing of the homicidal chauffeur problem where the evader has mostly constant

speed, a value of $\theta = 0.75$ is used.

$$e_n = \sum_{0}^{\infty} \{y_{n-r} - [p^*(r)]_n\}^2 \theta^r \tag{4.11}$$

$$g = 1 - \theta^3 \tag{4.12}$$

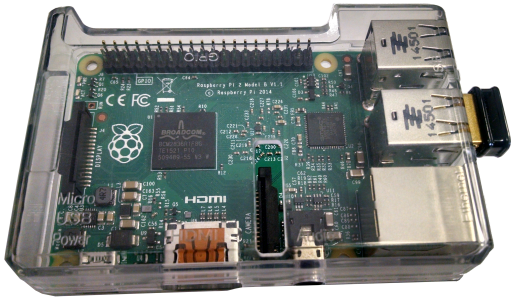
$$h = 1.5(1 - \theta^2)(1 - \theta) \tag{4.13}$$

$$k = 0.5(1 - \theta)^3 \tag{4.14}$$

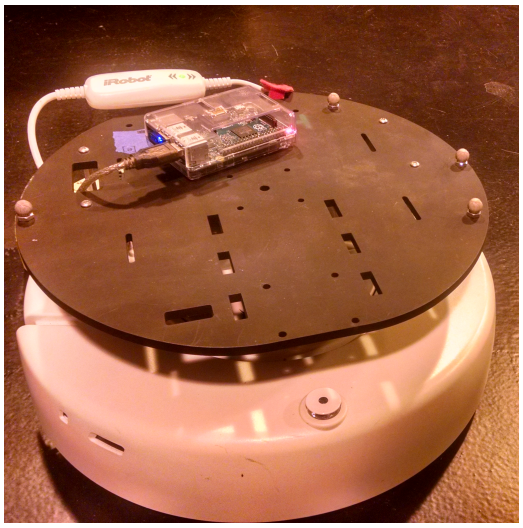
The choice of the STEP to represent the evader is twofold. First, it can achieve the same dynamics as the evader, which the iRobot® Create cannot do. To achieve instantaneous heading changes with Create robot, the experiment run-time and the simulation run-time must be two separate domains, where the simulation time effectively stops and starts to accommodate the true turn of the robot. This was tested early on and demonstrated as a sufficient if somewhat convoluted way to achieve instantaneous heading change on a non-holonomic vehicle. The STEP can change direction of motion instantly, so there is no need to accommodate special behavior or timing.

Secondly, for the purpose of the demonstration and capture within a small radius, the STEP is again the better platform. Using the STEP, a small evader plate can be mounted at the bottom of the pendulum at any height above the ground. Since the PE game is played on an open 2D plane, only x and y coordinates are required and any z offset in the evader plate is ignored by any algorithm or sensing. The two vehicles can simulate a capture or rendezvous within a smaller physical region by being at this different height, whereas two identical ground robots can get no closer than $2r$, where r is the radius of the robot.

The pursuer vehicle (Create robot) has two computers on board. The first is an internal black box hardware driver. This computer recognizes pre-defined function



(a) Raspberry Pi 2



(b) Connected to the iCreate robot

Figure IV.4: Raspberry Pi 2 integration with the iCreate robot ground vehicle

commands and passes them to the robot's motors. The interface between serial and USB that the robot provides allows commands from an external source to be defined and communicated as well. The second computer used is the Rasberry Pi™ 2 Model B. The Rasberry Pi (RasPi) is a low-cost ARM linux platform featuring a single-board CPU developed and marketed for educational use and outreach. Here, it is useful as a simple and easily-programmable computer that can be integrated well with the Create robot. The RasPi 2 Model B has a Broadcom BCM2836 processor which runs ARMv7 at 900MHz. Other features of the board include a dual core graphics card, 1 Gb memory, SD-card boot capability, several USB sockets, ethernet connection, and a dedicated 40-pin GPIO board. Details of these specifications can be found online and are comparable to other similar single-board computers like the Odroid [30, 31]. Figure IV.4 depicts the RasPi 2 with a WiFi dongle and bootable SD card inserted.

All coding for the hardware experimentation and real-time solution of the work

herein was done in C++. The RasPi computer was installed with a lightweight version of Ubuntu, Xubuntu, which runs on a simple linux architecture. This was done largely to facilitate the incorporation and use of the Robot Operating System (ROS). ROS is an open-source and open-contribution C++ software suite. It consists of many packages created specifically for simplifying the control of basic hardware test-beds and commercial robots. ROS is frequently used in academia to seamlessly transfer from simulation to testing of hobbyist ground vehicles such as the Create or Turtlebot.

For this work, several ROS packages were written and used. The unique contribution of ROS is a simple communication protocol that works by managing "publishers" and "subscribers" - objects that want to send or receive messages containing various specific data types. ROS is used as the handler to broadcast these messages. For example, when a Vicon measurement is taken by the MoCap system, it needs to be read by the various vehicles trying to access that measurement. To do this, a ROS subscriber object is created that subscribes to the IP address of the Vicon data stream. In this manner, measurements, states, and other commands can be shared from sensing devices to the vehicles and even between the vehicles.

IV.D. System Integration

The proposed integrated system is shown in Fig. IV.5. In the diagram, Vicon streams truth data about each vehicle's state to its computer. This data can be manually corrupted to include noise, or can be used to replicate knowledge of one's own state. Noisy measurements of another vehicle's state can be sent between vehicles. As each vehicle's computer receives state information, it computes the guidance and control laws required and passes those commands to the desired actuator or motor.

The experiment, or game, will end when the Vicon truth data indicates that the terminal condition has been reached.

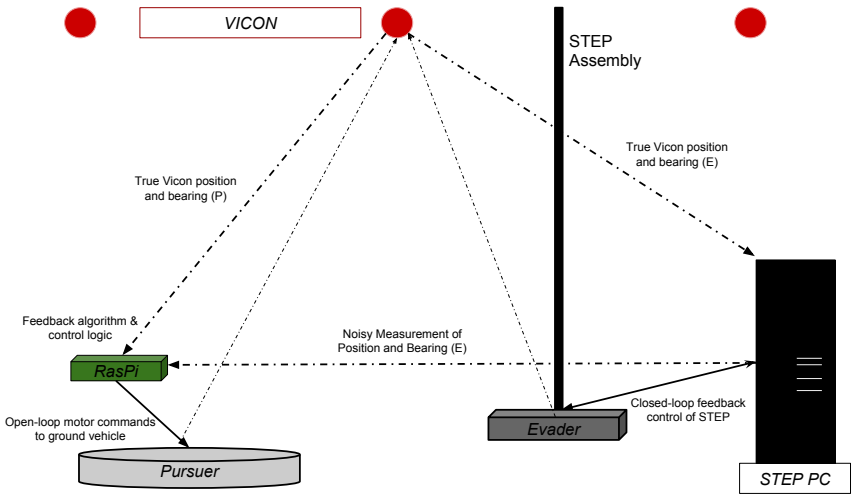


Figure IV.5: System diagram of homicidal chauffeur demo

IV.E. Experiment

For HWIL experimentation, it was desired to replicate the same scenarios as were presented in Chapter III inside the laboratory space. The four test scenarios are repeated here[1].

1. A passive ZOH evader runs at a heading of 0.00 rad

2. An optimal ZOH evader runs at a heading of 0.31 rad

[1]For real-time testing, initial conditions differ from earlier simulation due to spatial and vehicle constraints. Given the new initial conditions, the optimal ZOH evader heading becomes $\theta_e^* = 0.31$ rad

3. A passive evader runs at a heading of 0.00 rad, then increases speed and switches to an optimal heading of 0.31 rad

4. An optimal evader runs at a heading of 0.31 rad, then switches to a suboptimal heading of -0.80 rad with no speed change

To accommodate the physical constraints of the test space and vehicles, the values of several simulation variables were changed. Further, the planning horizon and control bandwidth were both decreased to improve speed on hardware. These changes had little to no effect in simulation. See Table IV.1 for details. To evaluate the

Table IV.1: Variables and their simulation and experiment values

Variable	**Simulation Value**	**Experimented Value**
v_p	1.00	0.25
v_e	0.40	0.10
r_p	0.5	0.25
R_0	$\sqrt{5}$	$\sqrt{1.25}$
R_{cap}	0.8	0.6
t_H	2.0	1.0
t_E	0.05 (20 Hz)	0.10 (10 Hz)

performance of the controller on hardware, two metrics were used. First, the time of capture t_f was compared between simulation and the laboratory. Second, the trajectory difference was calculated as a relative error in the distance (R) and heading (θ_p) of the pursuer.

IV.F. Experiment Results

IV.F.1. ZOH Evader Heading

In case 1, the test replicates a ZOH evader heading at $\theta_e = 0$. In case 2, the experiment replicates a ZOH evader heading at $\theta_e^* = 0.31$. Figures IV.6-IV.9 showing trajectory comparisons, position-based error, and heading error of the pursuer are below.

It is seen that the relative error in R grows over time. This can be explained by the lack of modeling of vehicle inertias, disturbances, and simplifying assumptions. If there is a small constant error in R, then as the reference value decreases, the relative error will increase. Since the duration of the scenarios in the scope of this work is short and the capture radius is relatively large, the position error stays within about 5% of simulation values. Similarly, the capture time differs by about 5-6% from simulation to experiment. Again, this is very good agreement considering the model and environment ignorance. The biggest contributor to error and LO is the heading error of the pursuer vehicle during the experiment, which is caused by controller tolerance, real-time effects, and vehicle fidelity. The Create is a simple COTS vehicle, and neither its model nor controller account for inertias, wheel slip or friction. The controller tolerance could also be tuned better, so that less heading error is allowable or oscillatory controls are eliminated.

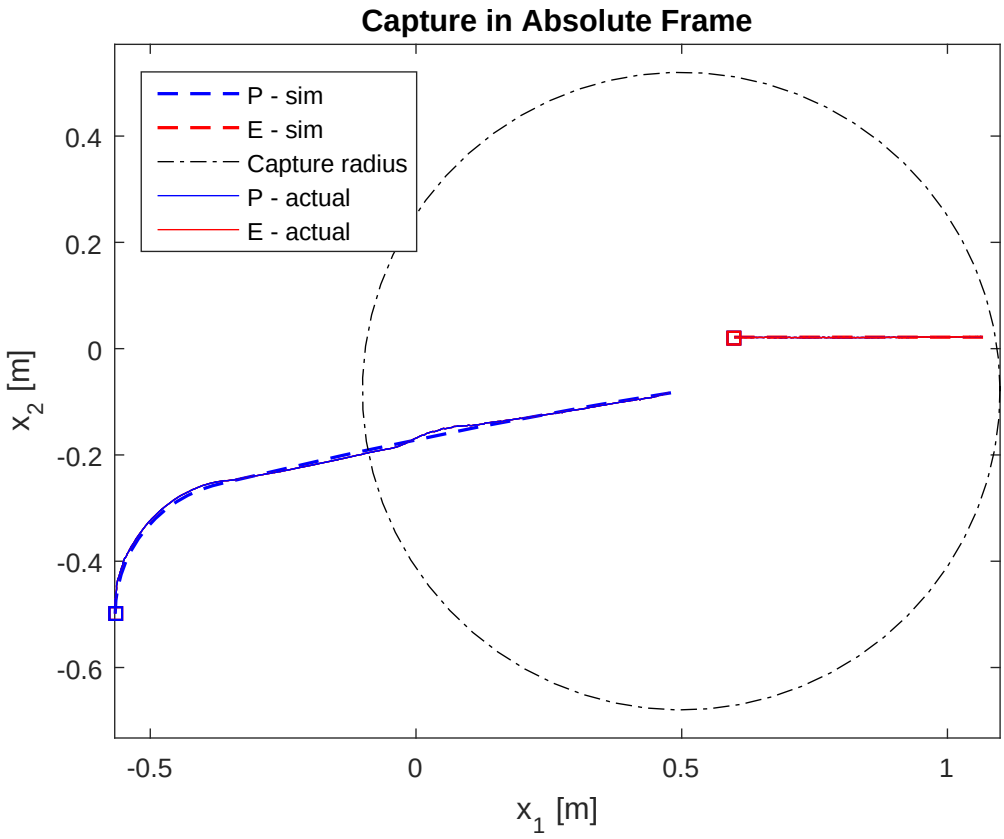
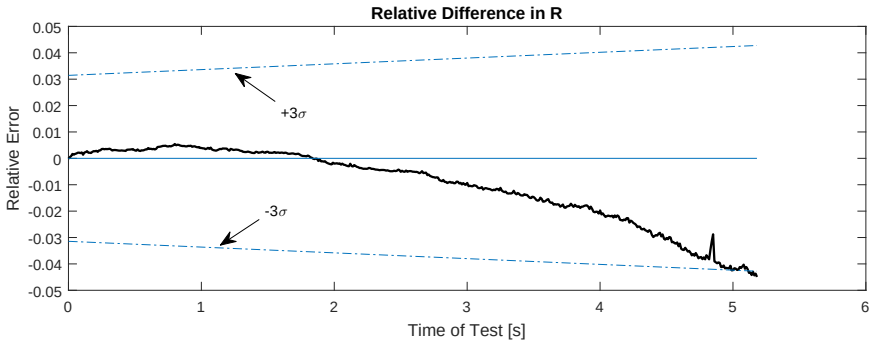
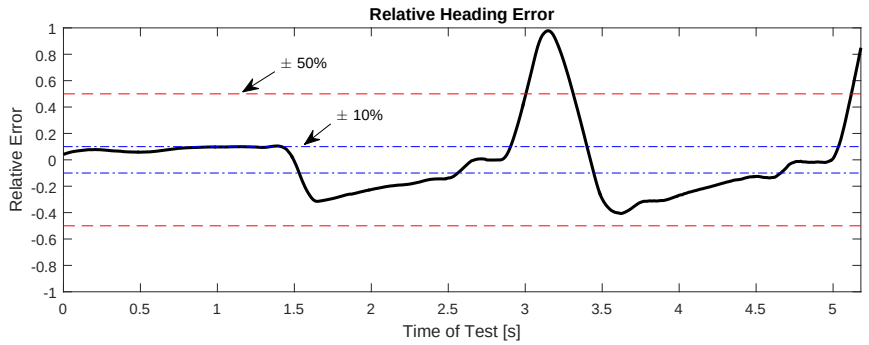


Figure IV.6: Phase plane overlay of actual results on simulation trajectories; Case 1



(a) R error



(b) θ_p error

Figure IV.7: Relative error in pursuer distance R and heading θ_p; Case 1

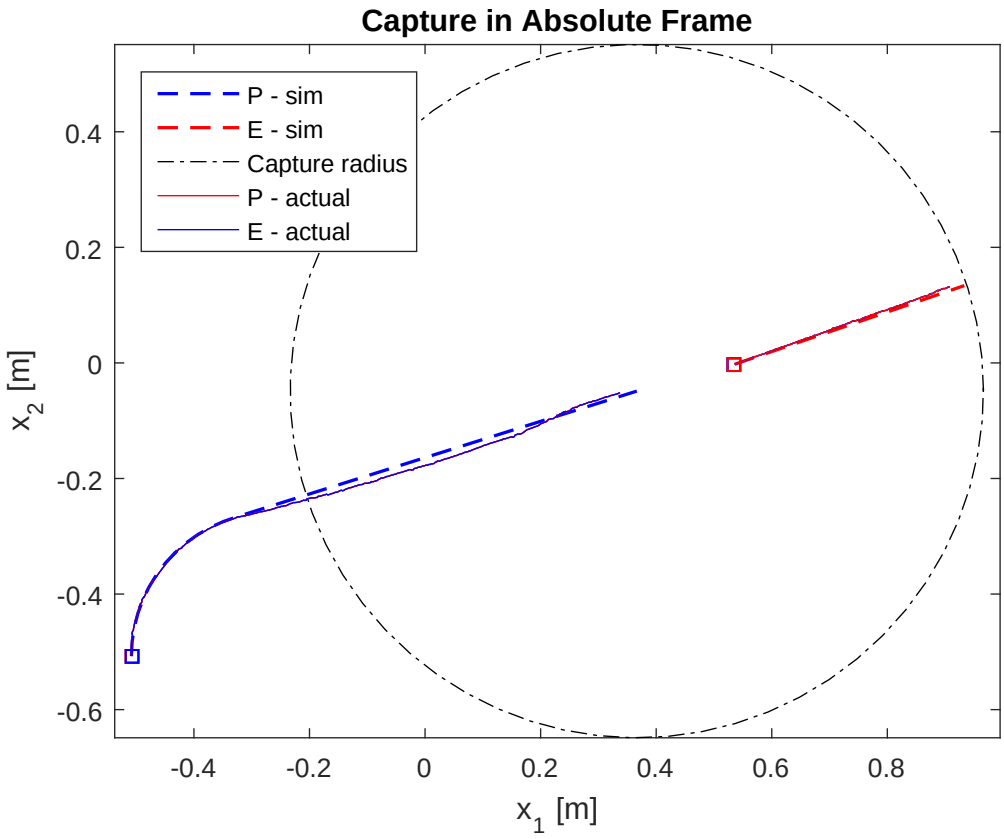
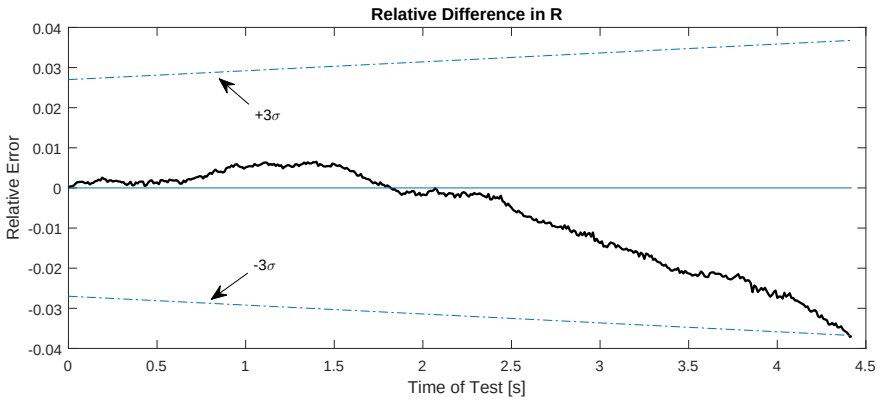
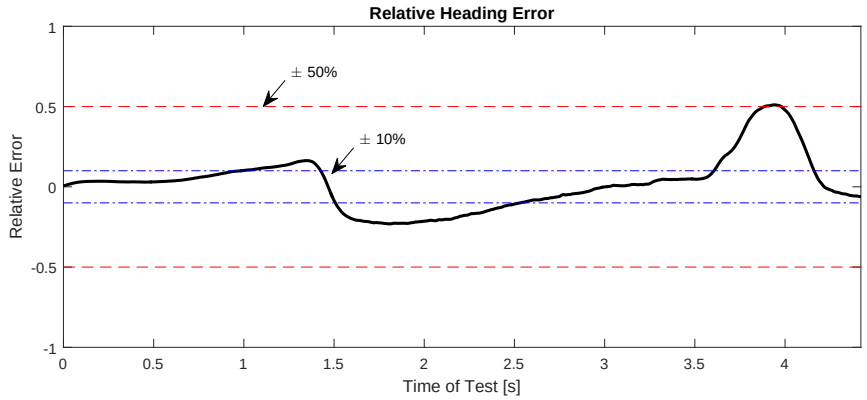


Figure IV.8: Phase plane overlay of actual results on simulation trajectories; Case 2



(a) R error



(b) θ_p error

Figure IV.9: Relative error in pursuer distance R and heading θ_p; Case 2

IV.F.2. Dynamic Evader Heading

In case 3, the test replicates an evader heading initially at $\theta_e = 0$ rad, then switching to an optimal trajectory and speeding up. In case 4, the experiment replicates an evader heading initially at $\theta_e^* = 0.31$, then switching to a suboptimal heading. Again, Figs. IV.10-IV.13 show trajectories and relative errors. As seen in Fig. IV.10, contributors to error are the true STEP heading and the RHC controller tolerance ξ. Figure IV.12 indicates that ξ and a bad prediction cause error in case 4.

Once again, the relative error in R grows over time. The position error stays within 1-3% of simulation values, and heading error varies more dramatically. The LO from simulation to hardware is roughly 8% in cases 3 and 4. This is still good agreement considering the model and environment ignorance. The capture times for all four cases are given in Table IV.2.

Table IV.2: Capture times

Case (Run)	**Simulation t_f**	**Experimental t_f**	**LO**
1	4.885 s	5.177 s	5.98%
2	4.170 s	4.420 s	6.00%
3	4.813 s	5.215 s	8.35%
4	3.452 s	3.750 s	8.63%

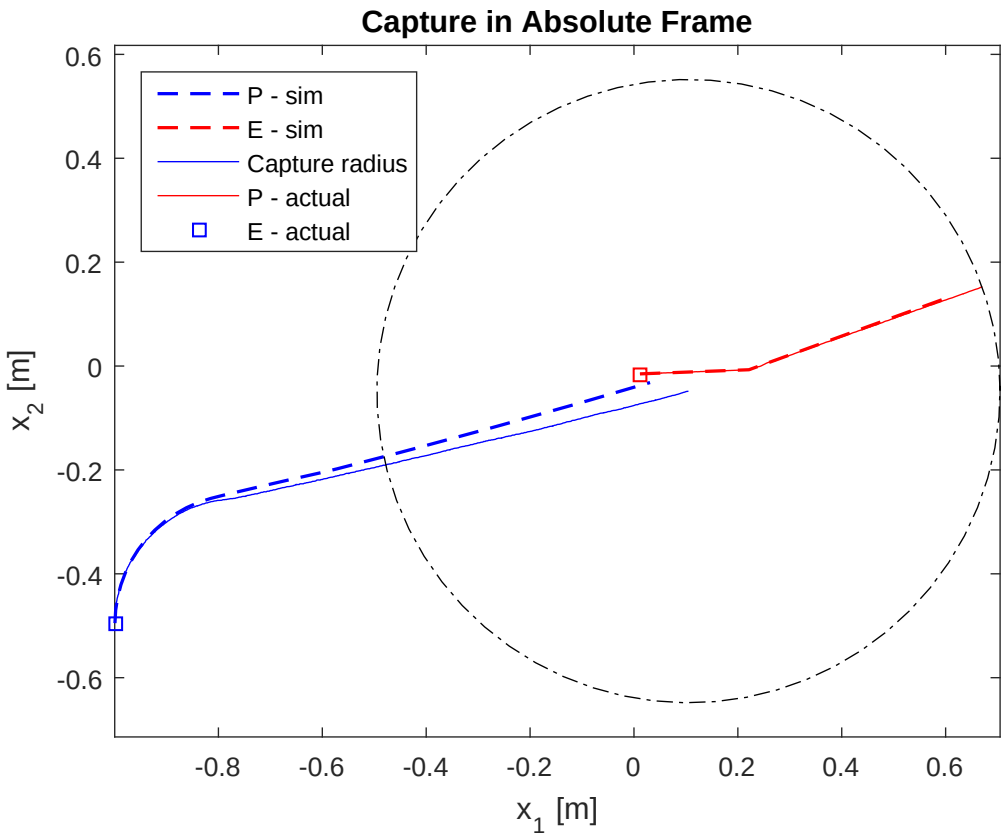
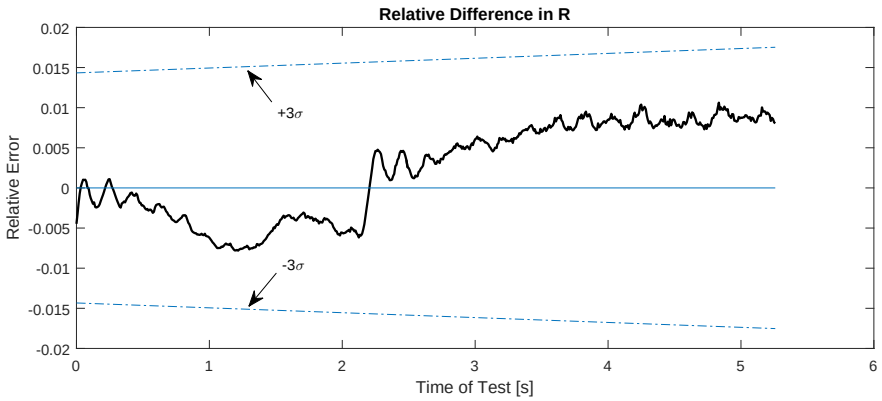
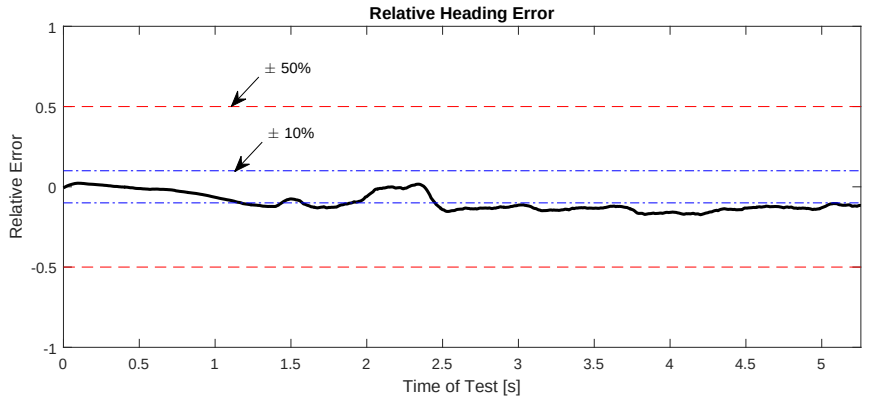


Figure IV.10: Phase plane overlay of actual results on simulation trajectories; Case 3



(a) R error



(b) θ_p error

Figure IV.11: Relative error in pursuer distance R and heading θ_p; Case 3

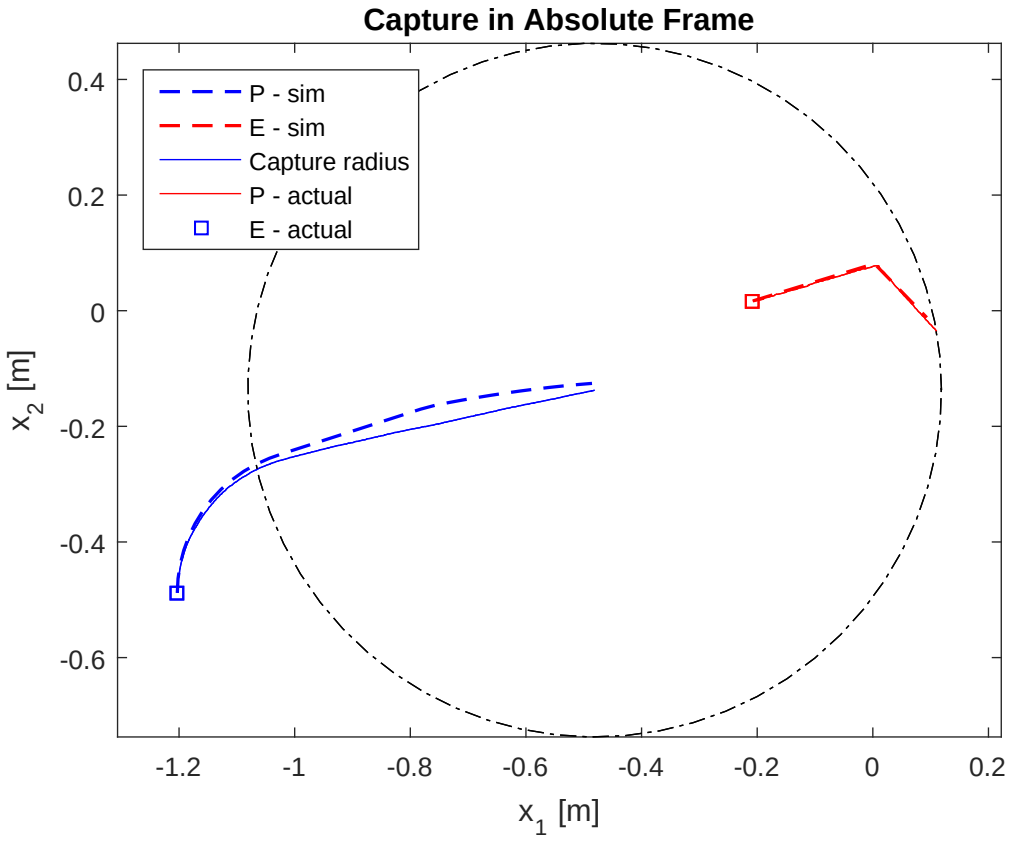
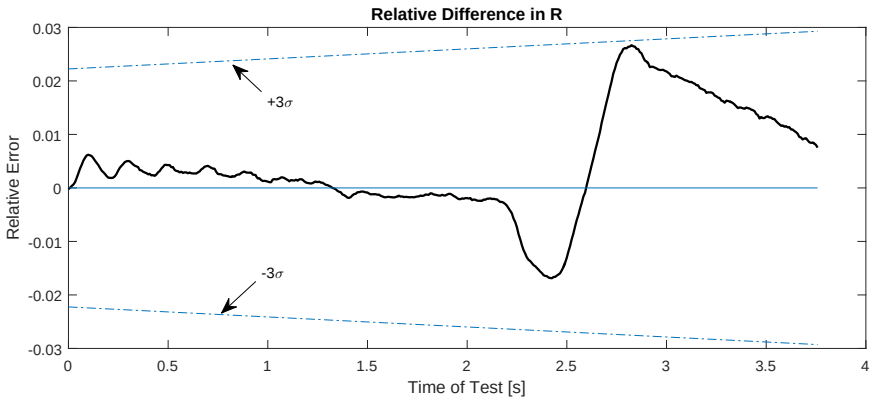
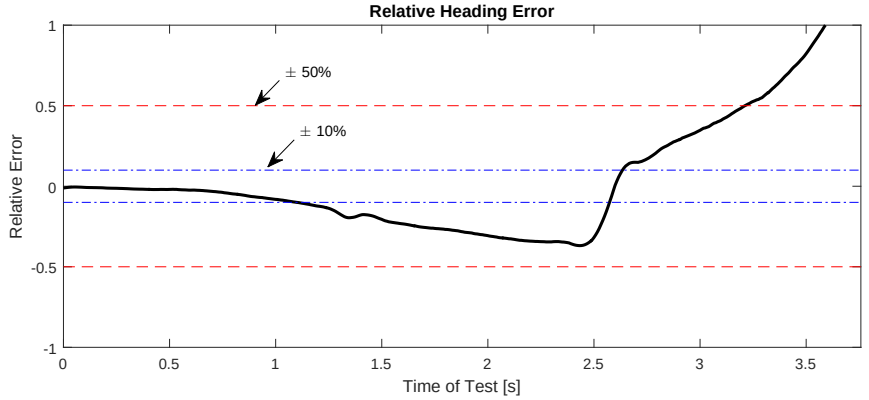


Figure IV.12: Phase plane overlay of actual results on simulation trajectories; Case 4



(a) R error



(b) θ_p error

Figure IV.13: Relative error in pursuer distance R and heading θ_p; Case 4

CHAPTER V
CONCLUSION

V.A. Summary

In this thesis a receding horizon controller was developed for nonlinear pursuit-evasion games with uncertainty. The RHC approach with an assumption of a passive evader was shown to be more effective with incorrect strategy estimates than the game-optimal control method. The RHC algorithm can be solved for nonlinear systems using numerical minimization over a finite horizon or by making numerical approximations. The homicidal chauffeur game was used as an example system for control implementation and testing. The game dynamics exhibit time-scale separation in the pursuer's motion and so a singular perturbation approach can be taken to approximate the optimal minimum-time solution. This SPT approach was shown to yield performance equal to the numerically minimized solution at each time step. With practical sampling and planning horizons, the RHC algorithms yielded results within 1% LO of the optimal full-knowledge solution, even with incorrect estimates of the evader's behavior.

The receding horizon algorithms were validated both in real-time simulation and on simple hardware. An online autonomous version of the homicidal chauffeur game was played out for several test cases and matched simulation results well. The cost function $J_p = t_f$ differed by $< 9\%$ and other performance metrics indicated equally good agreement. All hardware testing was done without modeling vehicle inertias or frictions and without explicitly accounting for disturbances that are always present in real scenarios. With added modeling and consideration of disturbance characteristics, the alignment between results would likely be even better. Much of

the error seen in hardware testing can be attributed to vehicle control precision and to noise characteristics of the rigid STEP assembly. It may be possible to better cancel the oscillatory modes of the assembly with additional suspension support. This is the first known online/onboard implementation of the homicidal chauffeur game in hardware, and the method yields a system that is stable and robust to unmodeled additive disturbance.

Although the SPT is based on the assumption that $r_p/R_0 \ll 1$, it may not hold true as the game is played out. As the vehicles approach each other, r_p/R_0 will continue to increase. There is certainly LO present due to this characteristic, but the algorithm still performs extremely well and total LO remains within 1-5%. It is unlikely it would continue to do so if the terminal inequality constraint (capture region) became a terminal equality constraint (point capture).

V.B. Future Work and Applications

The RHC presented in this work avoids behavior-learning techniques that can be a computational burden, especially for nonlinear systems. However, it would be useful to implement an adaptive estimator (filter) to tune the length of the planning horizon t_H in real time. For example, an adaptive t_H could vary in length based on the opposing vehicle's most recent states and a calculated likelihood of a ZOH. The idea is that if a target is rapidly maneuvering, the estimator would determine that the likelihood of a long duration of straight flight (ZOH) is small, and hence decrease the planning horizon length. Conversely, if the target has not maneuvered in several samples, the confidence in a passive or ZOH control strategy would increase and the controller may begin to increase the planning horizon of the RHC algorithm. This estimator could run in parallel to the control law and simply return running

estimates of the best t_H to use with optimization.

To increase the fidelity of the results given here, additional nonlinearities, disturbances, and actuators could be incorporated into the vehicle models and game. The game could also become more interesting if played in a higher-dimensional space rather than on a planar surface with ground vehicles. By using a stochastic problem formulation and output feedback, one could continue to make strides toward realistic hardware scenarios.

Finally, the RHC approaches here may be applied to more application-based scenarios. For example, a current research interest at the LASR Lab is rendezvous between a chaser vehicle and defunct space debris for debris-removal purposes. After reformulating the cost function to account for spacecraft fuel and proper positioning of capture mechanisms, a receding horizon approach could be employed to estimate a simplified tumbling motion of the debris body and optimize an online approach and capture path.

REFERENCES

[1] Isaacs, R., "Games of Pursuit," Tech. rep., RAND Corporation, Santa Monica, 1951.

[2] Isaacs, R., *Differential Games*, NY: John Wiley, 1965.

[3] Starr, A. and Ho, Y., "Nonzero-Sum Differential Games," *Journal of Optimization Theory and Applications*, Vol. 3, No. 3, 1969.

[4] Bryson, A. and Ho, Y., *Applied Optimal Control*, Ginn and Company, 1969.

[5] Shinar, J. and Glizar, V., "Application of Receding Horizon Control Strategy to Pursuit-Evasion Problems," *Optimal Control Applications and Methods*, Vol. 16, 1995, pp. 127–141.

[6] Copp, D. and Hespanha, J. P., "Nonlinear Output-Feedback Model Predictive Control with Moving Horizon Estimation," *Proc. of the 53nd Conf. on Decision and Contr.*, December 2014.

[7] Copp, D. and Hespanha, J. P., "Nonlinear Output-Feedback Model Predictive Control with Moving Horizon Estimation: Illustrative Examples," Tech. rep., Center for Control, Dynamical Systems, and Computation; UC-Santa Barbara, October 2015.

[8] Cavalieri, K., *Incomplete Information Pursuit-Evasion Games with Applications to Spacecraft Rendezvous and Missile Defense*, Ph.D. thesis, Texas A&M University, 2014.

[9] Cavalieri, K., Satak, N., and Hurtado, J., "Incomplete Information Pursuit-Evasion Games with Uncertain Relative Dynamics," *AIAA SciTech - AIAA Guidance, Navigation and Control Conference*, January 2014.

[10] Lewis, F. L. and Syrmos, V. L., *Optimal Control*, Wiley Interscience, 1995.

[11] B. Houska, H. F. and Diehl, M., "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization," *Optimal Control Applications and Methods*, Vol. 32, No. 3, 2011, pp. 298–312.

[12] Houska, B., Ferreau, H., Vukov, M., and Quirynen, R., "ACADO Toolkit User's Manual," http://www.acadotoolkit.org, 2009–2013.

[13] Hargraves, C. R. and Paris, S. W., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," *Journal of Guidance, Control, and Dynamics*, Vol. 10, 1987, pp. 338–348.

[14] Enright, P. J. and Conway, B. A., "Discrete Approximations to Optimal Trajectories Using Direct Transcription and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992.

[15] Williams, P., "Direct Approaches for Solving Optimal Control Problems in MATLAB," Tech. rep., RMIT University, Melbourne, Australia, 2005.

[16] Simakova, E., "Differential Pursuit Game," *Automat. Remote Control*, Vol. 2, 1967, pp. 173–181.

[17] Tikhonov, A., "Systems of Differential Equations Containing Small Parameters in the Derivatives," *Mat. Sb. (N.S.)*, Vol. 31, No. 3, 1952, pp. 575–586.

[18] Shinar, J., "On Applications of Singular Perturbation Techniques in Nonlinear Optimal Control," *Automatica*, Vol. 19, No. 2, 1983, pp. 203–211.

[19] Calise, A. J., "Singular Perturbation Techniques for Flight-Path Control," *Journal of Guidance, Control, and Dynamics*, Vol. 4, No. 4, 1981, pp. 398–405.

[20] Michalska, H. and Mayne, D. Q., "Receding Horizon Control of Nonlinear Systems," *IEEE Transactions on Automatic Control*, Vol. 35, No. 7, 1990, pp. 814–824.

[21] Nicolao, G. D., Magni, L., and Scattolini, R., "On the Robustness of Receding-Horizon Control with Terminal Constraints," *IEEE Transactions on Automatic Control*, Vol. 41, No. 3, 1996, pp. 451–453.

[22] Jadbabaie, A. and Hauser, J., "On the Stability of Receding Horizon Control with a General Terminal Cost," *IEEE Transactions of Automatic Control*, Vol. 50, No. 5, 2005, pp. 674–678.

[23] Michalska, H. and Mayne, D. Q., "Robust Receding Horizon Control of Constrained Nonlinear Systems," *IEEE Transactions on Automatic Control*, Vol. 38, No. 11, 1993, pp. 1623–1633.

[24] Karafyllis, I. and Kravaris, C., "Global Stability Results for Systems Under Sampled-Data Control," *International Journal of Robust and Nonlinear Control*, Vol. 19, No. 10, 2008, pp. 1105–1128.

[25] Grimm, G., Messina, M. J., Tuna, S. E., and Teel, A. R., "Examples When Nonlinear Model Predictive Control is Nonrobust," *Automatica*, Vol. 40, No. 10, 2004, pp. 1729–1738.

[26] Gyurkovics, E. and Elaiw, A., "Stabilization of Sampled-data Nonlinear Systems by Receding Horizon Control via Discrete-time Approximations," *Automatica*, Vol. 40, 2004, pp. 2017–2028.

[27] Janisch, B., Hurtado, J., and Brink, K., "Numerically Approximated Receding Horizon Control for Uncertain Pursuit-Evasion Games," *Proc. of the 26th AAS/AIAA Space Flight Mechanics Meeting*, Vol. 158, American Astronautical Society, February 2016.

[28] NASA, "Technology Readiness Level," https://www.nasa.gov, 2015, accessed 2016-01-22.

[29] Brookner, E., *Tracking and Kalman Filtering Made Easy*, John Wiley and Sons, 1998.

[30] Adafruit Industries, "Raspberry Pi Product Detail Page," https://www.adafruit.com/products/2358, 2015, accessed 02/06/2016.

[31] Hardkernel co. Ltd., "ODROID Product Main Page," http://www.hardkernel.com/main, 2013, accessed 02/06/2016.

APPENDIX A

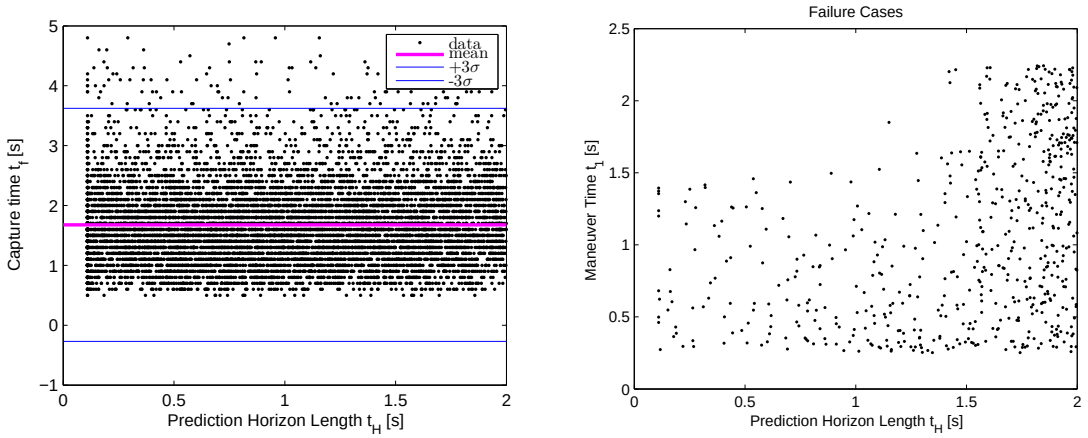
CONTROLLER STABILITY RESULTS

Investigation I

The initial Monte Carlo batch considers $n = 10000$ runs with the requirement that $t_H > t_E$ (fixed) and the initial evader positions are uniformly distributed in the vicinity of the pursuer's initial position at (0,0) according to Table A.1. In this and the following investigations, a Gaussian additive disturbance is applied to the pursuer's angular velocity. As shown in Fig. A.1a, the horizon length t_H has no statistical significance on capture time, provided that it is longer than t_E. Figure A.1b shows the failures under these conditions, as defined when the game simulation time exceeded $t_{limit} = 10$ s. Failure occurrences are skewed to long prediction horizons coupled with late maneuvers, where the evader makes a "last-ditch" velocity change.

Table A.1: Variables and distributions for MC simulation I

Variable	**Distribution**	**Description**
$x_{1_e}(0)$	$\sim \mathcal{U}(1, 2)$	Evader initial x_1 position
$x_{2_e}(0)$	$\sim \mathcal{U}(-2, 2)$	Evader initial x_2 position
θ_{e_0}	$\sim \mathcal{N}(0, \pi/3)$	Initial evader heading
θ_{e_1}	$\sim \mathcal{N}(\theta_{e_0}, \pi/2)$	Evader heading after time t_1
t_1	$\sim \mathcal{U}(0.25, 2.25)$	Evader maneuver time (switch heading and velocity)
Δv_e	$+0.10$	(constant) Evader velocity change after time t_1
t_H	$\sim \mathcal{U}(0.11, 1)$	Controller optimization or planning horizon
t_E	0.10	(constant) Control period or bandwidth
$d_{\dot{\theta}_p}$	$\sim \mathcal{N}(0, 0.01)$	Gaussian additive disturbance on pursuer angular velocity



(a) (b)

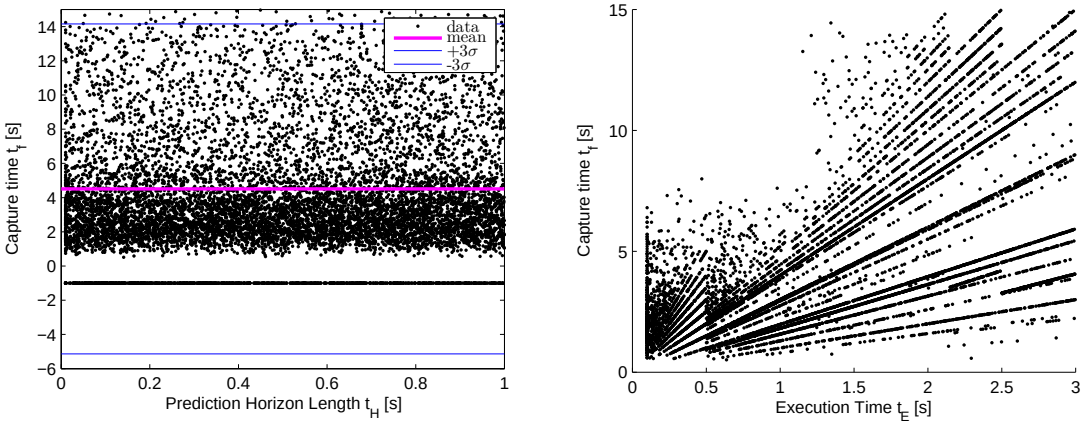
Figure A.1: MC simulation 1; (a) Capture time as a function of prediction horizon and (b) Failure cases

Investigation II

The second Monte Carlo batch considers $n = 10000$ runs with the allowance that t_E may be greater than t_H. Again, the distributions are given in Table A.2. The velocity increase after time t_1 was increased from +0.10 m/s to +0.25 m/s. The failure time t_{limit} was increased to 15 s to account for the larger Δv_e. A large t_E has much more effect on the capture time and possible failure of the game. A linear correlation between t_E and t_f can be seen in Fig. A.2b. The linearity comes from the discrete measurement and interpolation of capture times, nonetheless, the true trend is indeed that longer execution times decrease performance in a linear manner. It is also seen that all failures occur when $t_E > t_H$, and that many more failures occur at slow sampling (large t_E). This is shown in Fig. A.3.

Table A.2: Variables and distributions for MC simulation II

Variable	**Distribution**	**Description**
$x_{1_e}(0)$	$\sim \mathcal{U}(1,2)$	Evader initial x_1 position
$x_{2_e}(0)$	$\sim \mathcal{U}(-2,2)$	Evader initial x_2 position
θ_{e_0}	$\sim \mathcal{N}(0,\pi/3)$	Initial evader heading
θ_{e_1}	$\sim \mathcal{N}(0,\pi/2)$	Evader heading after time t_1
t_1	$\sim \mathcal{U}(0.25,2.25)$	Evader maneuver time (switch heading and velocity)
Δv_e	$+0.25$	(constant) Evader velocity change after time t_1
t_H	$\sim \mathcal{U}(0.01,1)$	Controller optimization or planning horizon
t_E	$\sim \mathcal{U}(0.10,3)$	Control period or bandwidth
$d_{\dot{\theta}_p}$	$\sim \mathcal{N}(0,0.01)$	Gaussian additive disturbance on pursuer angular velocity



(a) (b)

Figure A.2: MC simulation 2; Capture time as a function of (a) prediction horizon and (b) execution time

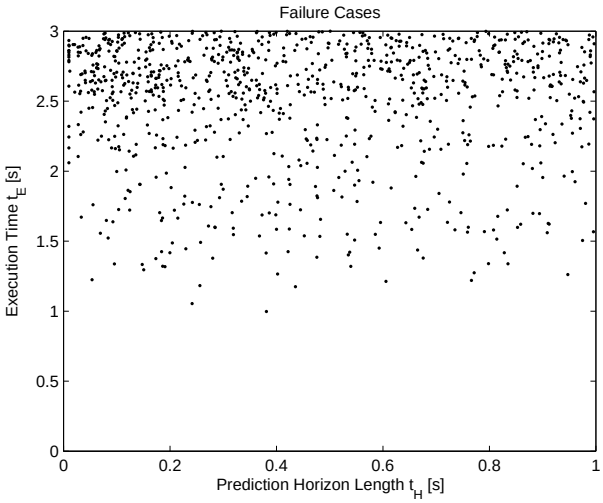


Figure A.3: MC simulation 2; Failure cases are skewed primarily toward longer execution times t_E and only occur when $t_E > t_H$

Investigation III

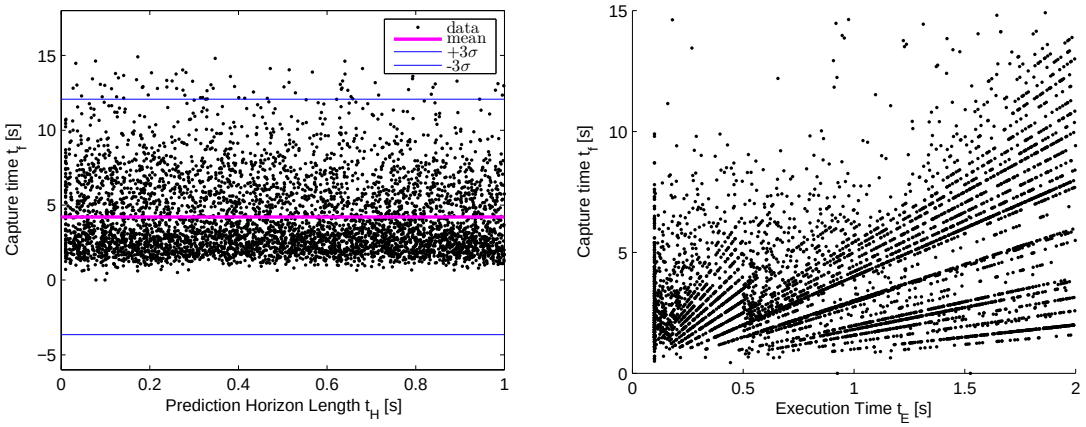
The final Monte Carlo investigation considers *two* batches of $n = 5000$ runs each with the allowance that t_E may again be greater than t_H. Here, the initial location of the evader relative to the pursuer is distributed differently to draw conclusions about capture dependence on initial LOS. In the first batch, $[x_{e_1}, x_{e_2}]^T \in \mathcal{S}$, where $\mathcal{S}$ is the set of points (x_1, x_2) that lie along an initial LOS $\psi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. In the second batch, $[x_{e_1}, x_{e_2}]^T \in \mathcal{S}^C$, where $\mathcal{S} + \mathcal{S}^C = \mathbb{R}$.

In both batches, t_f again depends on t_E. No explicit dependence is shown through t_H. While both batches show similar correlation between t_E and t_H, the failure cases differ between the two. When the initial evader position belonged to set $\mathcal{S}$, zero failures out of 5000 runs (0.00%) occured with t_E shorter than 1 s. Further, the total success of capture within t_{limit} was 98.52%. When the initial evader position belonged to $\mathcal{S}^C$, 47 failures occured with t_E shorter than 1 s (0.94%). The total failure count is more than twice that of the first batch, with an overall capture success of

95.42%. Therefore, one can conclude with confidence that at reasonable control bandwidth and an initial target position in set $\mathcal{S}$, capture is guaranteed.

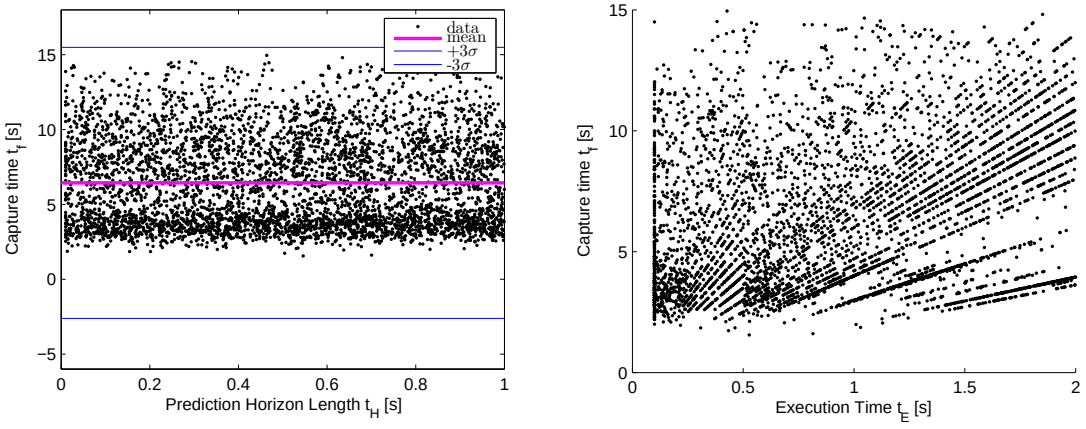
Table A.3: Variables and distributions for MC simulation III

Variable	**Distribution**	**Description**
$x_{1_e}(0)$	$\sim \mathcal{N}(0, 1)$	Evader initial x_1 position
$x_{2_e}(0)$	$\sim \mathcal{N}(\pm 3, 2/3)$	Evader initial x_2 position
θ_{e_0}	$\sim \mathcal{N}(0, \pi/3)$	Initial evader heading
θ_{e_1}	$\sim \mathcal{N}(0, \pi/2)$	Evader heading after time t_1
t_1	$\sim \mathcal{U}(0.25, 2.25)$	Evader maneuver time (switch heading and velocity)
Δv_e	$+0.25$	(constant) Evader velocity change after time t_1
t_H	$\sim \mathcal{U}(0.01, 1)$	Controller optimization or planning horizon
t_E	$\sim \mathcal{U}(0.10, 2)$	Control period or bandwidth
$d_{\dot{\theta}_p}$	$\sim \mathcal{N}(0, 0.01)$	Gaussian additive disturbance on pursuer angular velocity



(a) (b)

Figure A.4: MC simulation 3, batch 1; Capture time as a function of (a) prediction horizon and (b) execution time



(a) (b)

Figure A.5: MC simulation 3, batch 2; Capture time as a function of (a) prediction horizon and (b) execution time

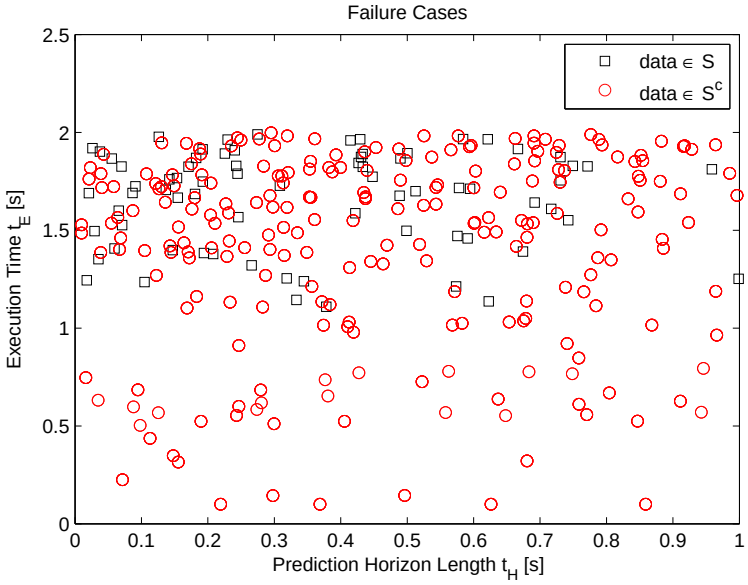


Figure A.6: MC simulation 3; Failure cases