Safety Control for Order Preserving Systems with Applications to Intelligent Transportation

by

Michael Robert Hafner

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in The University of Michigan
2012

Doctoral Committee:

Associate Professor Domitilla Del Vecchio, Co-Chair, MIT
Professor Jim Freudenberg, Co-Chair
Professor Jessy Grizzle
Professor Jing Sun

To my parents, family and Ally

ACKNOWLEDGEMENTS

It has been an honor to work with Domitilla. Her research interests and desire to impact the world through real-world engineering advancement has been quite the inspiration. It was not to long ago that I stumbled into her office as a first year graduate student with no research experience to speak of. Through meticulous and patient supervision, I had the opportunity to fail and conquer on many levels, for which I will be forever indebted. The close collaboration with Toyota provided an excellent opportunity to understand what it takes to change the world one controller at time. Everyone in the IVS group including Lorenzo, Drew, and Jeff were incredibly patient with my novice programming skills, and gave me room to work independently on fantastic state of the art engineering systems. In particular, I must thank Drew for the tireless hours he spent running my experiments on the check-road.

I will treasure the memories and friendship gained amongst my colleagues, both within the research group and outside, with whom I shared office space, laughs, burnt coffee and countless hours spent working on papers and homework. Group members from Michigan include Michigan Rajeev, Shridhar, and Mads; and the MIT contingent of Sid, Alessandro, Andras, Phillip and Reza. Other colleagues whom I shared office space include Koushil, Hongwei, Hae-Wan and Greg. I must also send my thanks to Shinung, who always served as a vast ocean of knowledge involving everything from qualifying exams to Michigan football. l enjoyed my time spent with friends from the Michigan auto-lab, which fielded one of the greatest D-League hockey teams Chelsea has and will ever see.

I would like to show my gratitude to the other individuals which have been important in my time at Michigan. The undergraduate students I had the opportunity to teach in discussion section and the undergraduate research student Jeff Duperret, who all kept me on my toes and introduced me to the wonderful world of teaching. The faculty members within our department and college provided excellent learning opportunities both in and away from the classroom. Becky was the greatest department assistant one could ever ask for, always responded promptly to email inquires and made paperwork fun. There were countless other students that provided the unique experiences in attending Michigan.

I am grateful for the emotional support of my family and loved ones, without which I would have not been able to finish. My mother provided frequent conversations of encouragement, and always offered assistance in financial matters. My father was the ever present backbone of the family, who always encouraged me to follow my dreams. I had the good fortune of living in Boston at the same time as my younger sister Kelly, who I have always admired for taking her own path through life. My brother Daniel was always fun to talk to, even with him living in Korea for the majority of my studies.

And lastly I must thank my wonderful Ally for all that she put up with. As possibly the most important gift I received while attending Michigan, her companionship has changed my life. We have developed many hobbies and pastimes together, and I have no doubt she will inspire me to continue to grow as a person. At many times, I considered leaving the program for personal reasons, and without her support I fear I would not have finished.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

CHAPTER I

Introduction

In this thesis, we address the problem of safety control with applications to vehicle collision avoidance. The contributions of this thesis are summarized below.

- An explicit representation of the capture set is derived for the two-agent safety control problem, under the assumption of imperfect state information and disturbance inputs, with significantly reduced complexity compared with the state of the art.
- A controller rendering the complement of the capture set controlled invariant is constructed based on the capture set representation, which can be computed in real-time using algorithms of linear complexity with respect to state.
- Our algorithms are applied both in simulation and on-board a multi-agent test-bed.
- These safety control algorithms are extended to a three-vehicle roundabout multi-agent test bed, where formal and experimental results guarantee non-blocking safety of the system.
- The assumptions needed for the safety algorithms are extended to include full size vehicle dynamics, communication delay and distributed control implementation.
- The algorithms are implemented on-board a full-size vehicle test-bed at the Toyota Technical Center, which demonstrates safety of our controller in a real-world setting.

1.1 Problem

The central problem addressed in this thesis is that of safety control for multi-agent systems. In the control literature, given a subset within the state space, this is traditionally called the computation of the maximal controlled invariant set for a dynamical system. This problem has the real-world interpretation of preventing collisions between mechanical agents in a distributed control topology. We look to extend state-of-the-art results to techniques that allow for implementation on-board real world-vehicles, where modeling can be imprecise and computation has limitations.

1.2 Motivation

In the United States, vehicular collisions kill on average 116 and injure 7,900 people [75] per day. In 2009, more than 33,800 people were killed in police-reported motor vehicle traffic crashes and about 2.2 million people were injured [9]. The estimated economic cost for all these police-report crashes was $230US billion. The National Safety Council says the odds of dying from a motor vehicle accident are 1 in 84, the fourth highest odds after heart disease, cancer, and stroke. The situation in the European Union is similar, with about 43,000 deaths and 1.8 million people injured per year, for an estimated cost of 160 billion euros [41]. In 2010, the number of fatalities from traffic accidents in Japan was 4,863. The total number of fatal and injury traffic accidents was 725,773. In April 2011, the Japanese government set the target to reduce fatalities to less than 3,000 by 2015 [8]. In 2009, light vehicle (passenger cars, sport utility vehicles, vans, and pickup trucks) crashes accounted for 68% of all U.S. motor vehicle fatalities and, of those light vehicle fatalities, 26% were from side impacts [9], suggesting crashes at intersections or on roadways close to and leading to intersections. In a different study from 1999 and 2000, 22% of all police-reported fatal accidents involving light vehicles occurred at traffic intersections.

These statistics clearly indicate that *crashes at traffic intersections have a major impact on the total amount of crashes and fatalities in the United States*. Furthermore, unlike other high-percentage crashes, such as road departure (23% of all crashes) and rear end (28% of all crashes), for which radar and camera-based forward collision systems are now available, there is currently no established technology to address side-impact collisions at intersections. Therefore, preventing unintentional stop-sign and red-light violations is one focus of this project.

1.3 Related Work

There two main domains of research literature we consider, is the control community concerned with safety control for general nonlinear systems, and the community involved with Intelligent Transportation Systems (ITS).

1.3.1 Safety Control of Hybrid Systems

In this paper, we consider a class of piecewise continuous systems that evolve on a partial order and propose an explicit solution to the two-agent safety control problem with imperfect state information.

There has been a wealth of research on safety control for general nonlinear and hybrid systems assuming perfect state information [71, 86, 91, 92]. In these works, the safety control problem is elegantly formulated in the context of optimal control and leads to the Hamilton-Jacobi-Bellman (HJB) equation. This equation implicitly determines the maximal controlled invariant set and the least restrictive feedback control map. Due to the complexity of exactly solving the HJB equation, researchers have been investigating approximated algorithms for computing inner-approximations of the maximal controlled invariant set [54, 55, 84, 92]. Termination of the algorithm that computes the maximal controlled invariant set is often an issue and work has been focusing on determining special

classes of systems that allow one to prove termination (see [86] and the references therein). The safety control problem for hybrid systems has also been investigated within a viability theory approach by a number of researchers (see [22, 43, 44], for example).

The above cited works focus on control problems with full state information and, as a result, static feedback control maps are designed. When the state of the system is not fully available for control, the above approaches cannot be applied. The advances in state estimation for hybrid systems of the past few years [10, 23, 24, 26, 31, 37, 85, 100] have set the basis for the development of dynamic feedback (state estimation plus control) for hybrid systems [34, 35, 102]. In particular, [102] proposes a solution to the control problem with imperfect state information for rectangular hybrid automata that admit a finite-state abstraction. For this case, the problem is shown to have exponential complexity in the size of the system. This problem is solved by determining the maximal controlled invariant safe set, that is, the set of all initial information states for which a dynamic control law exists guaranteeing that the current information state never intersects the set of bad states. Since the information state is a set, the maximal controlled invariant set is a *set of sets*, making its computation even harder than for the static feedback problem. As a consequence, for general hybrid systems the dynamic feedback problem under safety specifications is prohibitive. Dynamic feedback in a special class of hybrid systems with imperfect discrete state information is presented in [34], however the problem of computing the maximal controlled invariant set is not considered. Dynamic control of block triangular order preserving hybrid automata under imperfect continuous state information is considered in [35] for discrete-time systems, and an algorithm for computing an inner approximation of the maximal controlled invariant set is proposed. Dynamic feedback for order preserving systems in continuous time is considered in [36, 49]. However, in [36] only a cooperative game structure is considered and in [49] only a competitive game structure is addressed.

In [99], dynamic feedback is addressed for a class of hybrid automata with imperfect state information.

Since for general classes of hybrid systems, the dynamic feedback problem is prohibitive, we consider this problem in a restricted class of hybrid systems, which is still general enough to model application scenarios of interest. In particular, we focus on a class of hybrid systems whose state and input spaces have a partial ordering and generate trajectories that preserve this ordering. The problem is posed as an order preserving game structure, which is an approach that unifies the special cases of cooperative [36] and competitive [49] game structure between two agents in a general framework. By exploiting the order preserving property of the flow, we obtain an explicit solution for the maximal controlled invariant set and for a dynamic control map. We show that the static and dynamic feedback problems are solved by the same control map, which is computed on the state in the first case and on a state estimate in the second case. This implies separation between state estimation and control for the class of systems considered. For safety control problems generated by a specific conflict topology, this solution can be computed in discrete-time by linear complexity algorithms, for which we can show termination.

Dynamical systems whose flow preserves an ordering on the state space with respect to state and input are called monotone control systems [15]. Monotone control systems have received considerable attention in the dynamical systems and control literature as several biological processes involving competing or cooperating species are monotone [89]. More general bio-molecular systems can be modeled as the interconnection of monotone control systems [16, 17, 40]. There are also a large number of engineering applications that feature agents evolving on partial orders with order preserving dynamics. Multi-robot systems engaged in target assignment tasks have been shown to evolve according to an order preserving dynamics on the partial order established on the set of all possible assignments

[37]. Railway control networks feature a number of agents (the trains) that evolve on predefined paths (the railways) unidirectionally according to the Lomonossoff's model, which is an order preserving system on the path [58, 79]. Transportation networks also feature vehicles traveling unidirectionally on their paths and lanes, which impose an ordering on their motion. In air traffic networks, the longitudinal motion of each aircraft along its prescribed route can also be modeled by an order preserving dynamics [61, 82].

1.3.2 Intelligent Transportation Systems

In 2009, more than 33,800 people were killed in police-reported motor vehicle traffic crashes and about 2.2 million people were injured [9]. The estimated economic cost for all these police-report crashes was $230US billion. The situation in the European Union is similar, with about 43,000 deaths and 1.8 million people injured per year, for an estimated cost of 160 billion euros [41].In 2009, light vehicle crashes accounted for 68% of all U.S. motor vehicle fatalities and, of those light vehicle fatalities, 26% were from side impacts [9], suggesting crashes at intersections or on roadways close to and leading to intersections. These statistics clearly indicate that crashes at traffic intersections have a major impact on the total amount of crashes and fatalities in the United States. Furthermore, unlike other high-percentage crashes, such as road departure and rear end, for which radar and camera-based forward collision systems are now available, there is currently no established technology to address side-impact collisions at intersections. Therefore, preventing unintentional stop sign and red light violations has been subject of intense research in the past few years.

Vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) communication have been among the major technologies leveraged in research focused on preventing collisions at traffic intersections. In fact, the basic idea is that vehicles could cooperate with each other

and with the surrounding infrastructure sharing information about the environment to improve situational awareness. This would allow a vehicle to predict potential collisions that a driver may fail to foresee due, for example, to obstructed view or distraction. Based on this technology, intelligent transportation systems (ITS) for inter-vehicle cooperative safety continue to be examined world-wide by government and industry consortia, such as the Crash Avoidance Metrics Partnership (CAMP)[2, 3] and Vehicle Infrastructure Integration Consortium (VIIC)[5, 6] in the U.S., the Car2Car Communications Consortium in Europe [1], and the Advanced Safety Vehicle project 3 (ASV3) in Japan.

Previous work concerning formal approaches to ITS involved the automated highway systems (AHS) by the California PATH project in the 90s. The objective of the AHS project was to develop fully autonomous highway systems, with the goal of decreasing congestion, increasing safety, and improving fuel efficiency [52, 59, 81, 94]. Most of this work pertaining to safety involved the development of vehicle platooning, where formal modeling and control solutions were employed based on the computation the largest unsafe set, drawing from techniques in optimal control and game theory [12–14, 48, 60, 69, 70, 91]. Recent work concerning the design of intelligent intersections revolves around provably safe scheduling algorithms for large numbers of vehicles [62].

The employment of a formal hybrid modeling and control approach has been previously applied in the development of automated highway systems (AHS) by the California PATH project in the 90s [1]. The objective of the AHS project was the employment of fully autonomous highway systems, mainly based on the concept of platooning, to increase traffic throughput, safety, and fuel efficiency [52, 59, 81, 94]. In the context of platooning, a number of papers and PATH reports have proposed a formal hybrid modeling and control approach based on the computation of the safe set of initial conditions (the complement

[1] http://www.path.berkeley.edu/nahsc/default.htm

of the capture set), on optimal control, and on game theory [12–14, 48, 69, 70]. By contrast to the PATH project, we do not focus on fully autonomous highway systems, but on *partially autonomous* traffic intersection systems in which (i) not all vehicles approaching an intersection are assumed to be equipped with the on-board safety system (and thus they do not necessarily cooperate) and (ii) those vehicles that are equipped with the safety system are driven by humans, warnings are supplied when needed, and automatic control is applied only if necessary to prevent a collision. These two constraints are dictated by the application for a realistic deployment of this technology. From a theoretical standpoint, the partially autonomous nature of the system results in hybrid dynamical models, in which *the state of the system is not known* and thus it is not available to the controller. This structural feature renders the formal approaches previously investigated in the PATH project [12–14, 48, 69, 70] inapplicable as they assume perfect knowledge of the state of the system.

Due to the life-critical role of cooperative active safety systems, it is essential that these systems are designed so that they are guaranteed to be safe. This "guaranteed" design can be performed by employing formal methods, which have been investigated for a number of years both in the computer science and control communities [63, 83, 91, 92]. The control of agents under a safety specification can be addressed by computing the set of states that lead to an unsafe configuration independently of an input choice, called backward reachable set or capture set [91]. Then, a feedback is computed that guarantees that the state never enters such a set [39, 86]. To reduce the computational load, approximate algorithms have been proposed to compute an over-approximation of the capture set [54, 55, 92]. More recently, researchers have been tackling computational issues by focusing on restricted classes of systems (see, for example, [11, 45, 47, 50]).

Most of the cited approaches assume perfect knowledge of the system state. This as-

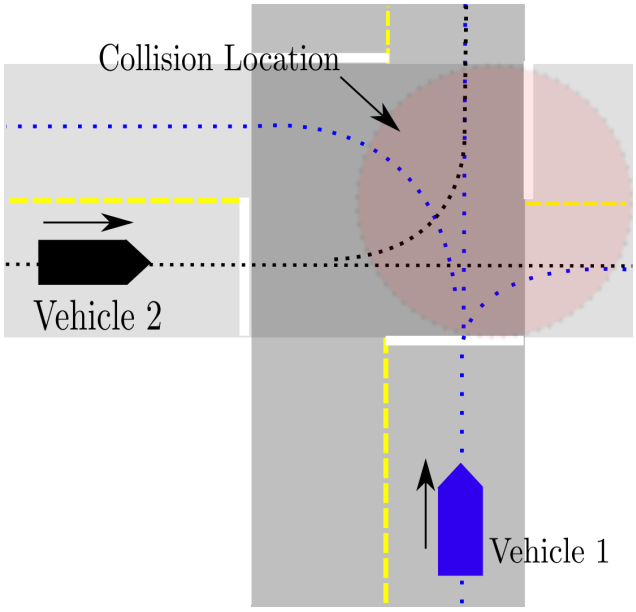


Figure 1.1: Example of two adjacent vehicles approaching a four way intersection. The location of potential collisions is shaded in red.

sumption is not satisfied in our application because of sensor noise and especially communication delays. We require algorithms that account for imperfect knowledge on the system state while being computationally efficient so that they can be implemented in real-time. Hence, in this paper, we apply the results of [50], which guarantee safety in the presence of imperfect state information, produce efficient algorithms suitable for real-time implementation, and only need a coarse model of the vehicle dynamics. Specifically, we focus on a two-vehicle collision avoidance scenario at a traffic intersections (Figure 1.1) and develop a decentralized control algorithm that uses V2V communication to determine whether automatic control is needed to prevent a collision. Here, we consider preventing a collision through automatic control by actuating only brake and throttle, but not steering, and assuming drivers follow nominal paths as established by the driving lanes. In our intersection collision avoidance (ICA) application, the drivers retain full control of the vehicle until the system configuration hits the capture set. At this point, a control action is necessary to prevent a collision, and automatic throttle or brake are applied to

both vehicles in a coordinated fashion so that one vehicle enters the intersection only after the other has exited it. After the crash has been prevented, the driver regains control of brake and throttle. We implemented our algorithms on two Lexus IS 250 test vehicles and performed a number of experiments with different use cases on a test intersection at the Toyota Technical Center of Ann Arbor, MI.

The employment of formal methods has been previously applied in the development of automated highway systems (AHS) by the California PATH project in the 90s. The objective of the AHS project was to deploy fully autonomous highway systems incorporating vehicle platoons to increase traffic throughput, safety, and fuel efficiency [52, 81]. In the context of platooning, a number of papers and PATH reports have proposed formal approaches based on the computation of the capture set, on optimal control, and on game theory [12, 70]. Recent work concerning centralized control of autonomous traffic intersections employs scheduling techniques to determine a possible safe trajectory [62] or to construct a safe supervisor that overrides the driver when a crash is imminent [30]. Similarly, centralized cooperative vehicle intersection control algorithms to enforce safety based on optimal control have appeared [66]. Experimental works have also recently appeared on full scale vehicle test-beds on collision avoidance systems at traffic intersections, which leverage V2V communication. In particular, in [72] a fuzzy controller to manage vehicles crossing an intersection is proposed, however safety guarantees are not provided. In [73], an on-board vehicle hazard detection that uses V2V is developed to warn the driver about dangerous situations. However, no automatic control is employed and no formal safety guarantees are provided. In this paper, we bridge the gap between formal methods and cooperative collision avoidance systems at traffic intersections by developing/testing an experimental cooperative collision avoidance system based on formal control theoretic techniques.

General control design problems under language specification (safety, for example) [63, 71, 80, 91, 92] have been extensively studied for discrete systems in the computer science literature (see [90] for an overview). A control perspective in the context of discrete event systems was given by [83]. The approach has been extended to specific classes of hybrid systems such as timed automata [18, 57] and rectangular automata [101]. For these classes of hybrid systems, implementation results using tools such as [54] showed that in practice the synthesis procedure is limited to control problems with a small number of control modes. Most of the work on safety controller design for general classes of hybrid systems has been concerned with the computation of reachable sets (see for example [71, 91, 92], and the references therein). The control problem under safety specifications can be addressed by computing the set of states that lead to an unsafe configuration independently of an input choice (called the backward reachable set [91] or the uncontrollable predecessor [56] of an unsafe set). Then, a feedback is computed that guarantees that the state never enters such a set [39, 86]. As it appears from these previous works, computational constraints usually limit the system to four or five continuous variables and to two or three discrete states. To reduce the computational load, approximate algorithms have been proposed to compute an over-approximation of the backward reachable set of the unsafe set [54, 55, 92]. More recently, researchers have been tackling computational issues by focusing on restricted classes of hybrid systems. In [11, 47], hybrid systems whose continuous dynamics is linear time-invariant and discrete state switching is due to transition guards are considered. An over approximation of the reachable set is computed using simulation techniques over bounded time in [47] and by using zonotopes in [11]. In [77], a hybrid system is considered whose discrete state can switch due to discrete control, discrete disturbance and discrete human input. Hybrid reachability results are then utilized to create an invariance-preserving discrete event system abstraction of the so called hybrid

human-automation system.

However, all the above works are concerned with state feedback, that is, the state of the system is assumed to be available to the controller. Hence, they are not directly applicable to the problem investigated in this proposal, in which only imperfect/partial information on the state is available. Initial work in safety control of hybrid systems with imperfect or partial state information can be found in [102] and in the PI's previous work [33–35, 38]. In [102], a controller that relies on a state estimator is proposed for finite state systems. The results are then extended to control a class of rectangular hybrid automata with imperfect state information, which can be abstracted by a finite state system. The proposed algorithm has exponential complexity in the size of the system. In [34, 38], we have proposed a partial order approach for the design of computationally efficient state estimation and control algorithms. In such a work, only discrete dynamic feedback is considered. Also, no algorithm is provided to compute the capture set. Initial results for the efficient computation of an approximation of such a set for a special class of discrete time hybrid automata with imperfect continuous state information is proposed in [33, 35] and extended to continuous time models in [36, 46, 49, 51]. More recently, initial results to handle imperfect information on the mode of the system have appeared [95].

1.4 Organization

This thesis is organized into three main chapters. Chapter II introduces the formal problem in a rigorous fashion. Once the problem is formulated, an explicit representation of the capture set assuming imperfect information is derived, and formal proofs are provided. A controller is generated based on this representation, and formally verified to maintain invariance of the complement of the capture set, thus guaranteeing safety of the closed loop system. Algorithms are provided, which are formally shown to compute the capture set,

and subsequently implemented both in simulation and on-board the multi-agent test-bed at the University of Michigan.

In Chapter III, we extend these results to a working three-vehicle roundabout system. To accomplish this, we break the global safety problem into a set of localized two-vehicle collisions, and show that our results extend to such a system. The formal description of the results are necessarily extended to hold for an arbitrary collection of control modules. Experimental results are provided to validate the safety of this system.

In Chapter IV we start by reintroducing the safety control problem with the specific goal of designing a full-scale vehicle safety controller. The assumptions are extended to the full vehicle case, which includes system identification, estimator design, and algorithmic extensions to handle distributed implementation of the controller. The system is validated by comprehensive experimental testing, where we successfully maintain safety in a variety of conditions.

CHAPTER II

Theory: Safety Control for Piecewise Continuous Systems on a Partial Order

This chapter addresses the two-agent safety control problem for piecewise continuous systems with disturbances and imperfect state information. In particular, we focus on a class of systems that evolve on a partial order and whose dynamics preserve the ordering. While the safety control problem with imperfect state information is prohibitive for general classes of nonlinear and hybrid systems, the class of systems considered in this paper admits an explicit solution. We compute this solution with linear complexity discrete-time algorithms that are guaranteed to terminate. The proposed algorithms are illustrated on a two-vehicle collision avoidance problem and implemented on a hardware roundabout test-bed.

2.1 Introduction

We consider a class of piecewise continuous systems that evolve on a partial order and propose an explicit solution to the two-agent safety control problem with imperfect state information. We illustrate the application of the proposed technique to a two-vehicle collision avoidance problem as found in traffic intersections or modern roundabouts in the presence of modeling uncertainty, missing communication, and imperfect state information.

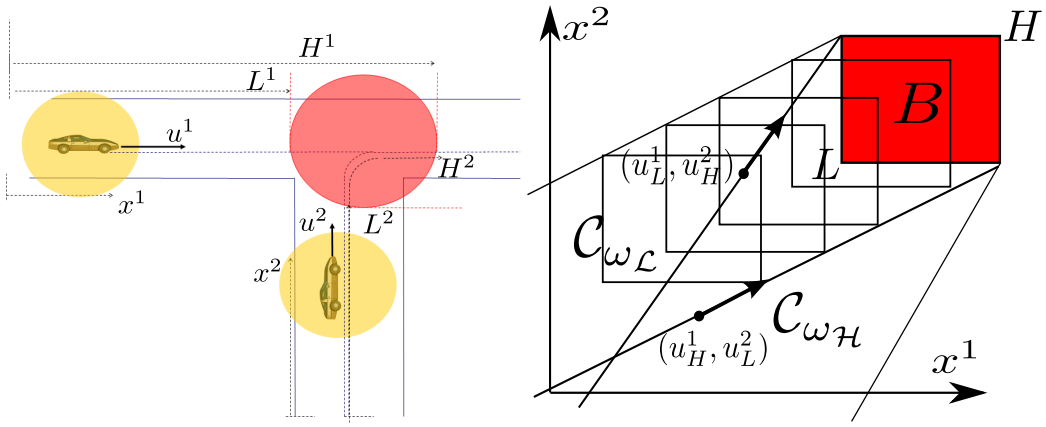


Figure 2.1: (Left) Vehicles approaching a "T" intersection. (Right) The sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$, in which $L = (L^1, L^2)$, $H = (H^1, H^2)$, and $B =]L^1, H^1[\times]L^2, H^2[$.

Motivating Example. Consider the problem of preventing a collision between two vehicles approaching an intersection as depicted in the left panel of Figure 2.1. A collision occurs if the two vehicles are in the shaded red area B at the same time. The problem is to design a controller that guarantees that the vehicles do not collide excluding the trivial solution in which the vehicles stop. In general, the vehicle states can be subject to large uncertainties as deriving from GPS, for example, and the dynamic model can be affected by modeling error. For the sake of explaining the basic idea of our solution, consider the case in which the dynamics of vehicles 1 and 2 are given by $\dot{x}^1 = u^1$, $\dot{x}^2 = u^2$, respectively, with $u^1, u^2 \in [u_L, u_H]$ and $u_L, u_H > 0$. A more realistic second order hybrid model for each of the vehicle dynamics will be considered in the simulation section. Assume also perfect information of the state (x^1, x^2). Here, x^1 and x^2 denote the longitudinal displacements of the vehicles on their paths as shown in the figure. In this coordinate system, $B =]L^1, H^1[\times]L^2, H^2[$. To solve the control problem, we seek to compute the set of all initial conditions $(x^1(0), x^2(0))$ that are taken to B for all inputs (u^1, u^2). This set is called the capture set, denoted C, and is the complement of the largest controlled invariant set that does not contain B. On the basis of the capture set, we then seek to design a feedback map that guarantees that any state starting outside C is kept outside C.

This general problem can be elegantly formulated as an optimal control problem with terminal cost, which leads to an implicit solution expressed as the solution of a PDE [71]. In this example, however, there is a rich structure that can be exploited to obtain an immediate explicit solution without the need of solving an optimal control problem. In particular, the dynamics of each vehicle preserve the standard ordering on $\mathbb{R}$, that is, higher initial conditions $x^i(0)$ and higher inputs u^i lead to higher values of the state $x^i(t)$ for all time. Denote by $C_{\omega_{\mathcal{H}}}$ the set of all initial conditions that are taken to B when the input to the system is set to $\omega_{\mathcal{H}} := (u_L, u_H)$, that is, vehicle 1 applies constant $u^1 = u_L$ and vehicle 2 applies constant $u^2 = u_H$. Similarly, denote by $C_{\omega_{\mathcal{L}}}$ the set of all initial conditions that are taken to B when the input to the system is set to $\omega_{\mathcal{L}} := (u_H, u_L)$, that is, vehicle 1 applies constant $u^1 = u_H$ and vehicle 2 applies constant $u^2 = u_L$. Because the dynamics of the system have the order preserving properties described above, one can show that the capture set is given by the intersection of these two sets, that is, $C = C_{\omega_{\mathcal{L}}} \cap C_{\omega_{\mathcal{H}}}$ (right panel of Figure 2.1). In practice, this means the following. The state x is taken to B for all input choices if and only if it is taken to B both when (a) vehicle 1 applies maximum control and vehicle 2 applies minimum control and (b) vehicle 1 applies minimum control and vehicle 2 applies maximum control.

The relevance of having $C = C_{\omega_{\mathcal{L}}} \cap C_{\omega_{\mathcal{H}}}$ resides in the following key points. First, $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$ can be easily computed by backward integration without the need of optimizing over the control values because the controls are fixed. Second, this backward integration task can be performed by simply propagating back through the dynamics the lower and upper bounds of B, that is, L and H, respectively, with fixed inputs (refer to the right panel of Figure 2.1). In discrete-time, this can be performed by a linear complexity algorithm. Furthermore, checking whether a state (x^1, x^2) belongs to either $C_{\omega_{\mathcal{L}}}$ or $C_{\omega_{\mathcal{H}}}$ can be performed in finite time because the backward integration of L and H leads to strictly de-

creasing sequences: once the decreasing sequences starting in H passes beyond the point (x^1, x^2), one has enough information to establish whether (x^1, x^2) belongs either to $C_{\omega_{\mathcal{L}}}$ or to $C_{\omega_{\mathcal{H}}}$. Finally, a feedback map is one that imposes the control $\omega_{\mathcal{H}} = (u_L, u_H)$ when the state is inside $C_{\omega_{\mathcal{H}}}$ and on the boundary of $C_{\omega_{\mathcal{L}}}$, while it imposes the control $\omega_{\mathcal{L}} = (u_H, u_L)$ when the state is inside $C_{\omega_{\mathcal{L}}}$ and on the boundary of $C_{\omega_{\mathcal{H}}}$ (right panel of Figure 2.1). This way, we provide also a closed form solution for the feedback map. *In this paper, we show that this basic result holds for arbitrary order preserving dynamics, for the case in which these dynamics are affected by disturbances, and when only imperfect state information is available*.

This chapter is organized as follows. In §2.2, we introduce basic definitions and the class of systems that we consider is introduced in §2.3. In §2.4, we provide a mathematical statement of the safety control problem. In §2.5, we give the main result of the paper, namely the computation of the maximal controlled invariant set and the related dynamic feedback control map. In §2.6, we present a discrete-time algorithm for computing the maximal controlled invariant set and the dynamic feedback map. In §2.7, we present an example application involving a two-vehicle collision avoidance problem at a traffic intersection. Several of the proofs are found in the Appendix.

2.2 Notation and Basic Definitions

For the set $A \subset X$ with X a normed vector space, denote the complement $\sim A := X \backslash A$, the interior $\mathring{A}$, the closure $\overline{A}$, the closed convex hull $\overline{\text{co}}\, A$, the boundary ∂A, and the set of all subsets contained in A by 2^A. For $x \in \mathbb{R}^n$, denote the Euclidean norm $\|x\|$, and the inner product $\langle y | x \rangle$. For $x \in \mathbb{R}^n$ and set $A \subset \mathbb{R}^n$, denote the distance from x to A as $d(x, A) := \inf_{y \in A} \|x - y\|$. This extends to the distance between two sets $A, B \in \mathbb{R}^n$, where $d(A, B) := \inf_{y \in A} d(y, B)$. Let $]a, b[$, $]a, b]$, $[a, b] \subset \mathbb{R}$ denote the open, half open, and

closed intervals respectively. This notation extends to interval sets $]a,b[,\ ]a,b],\ [a,b] \subset \mathbb{R}^n$, where, for example, $]a,b[:=]a_1,b_1[\times \ldots \times]a_n,b_n[$. The open ball of radius $\epsilon > 0$ centered at $x \in \mathbb{R}^n$ is denoted $B(x,\epsilon) := \{z \in \mathbb{R}^n \mid \|x-z\| < \epsilon\}$. For the set $A \subset \mathbb{R}^n$, we define an open neighborhood about A of radius $\epsilon > 0$ by $B(A,\epsilon) := \{z \in \mathbb{R}^n \mid d(z,A) < \epsilon\}$. Denote the canonical basis vectors $\hat{e}_i$ for $i \in \{1,2,\ldots,n\}$. For $x \in \mathbb{R}^n$, denote the i^{th} component by $x_i := \langle x | \hat{e}_i \rangle$. Denote the canonical projection $\tau_i : \mathbb{R}^n \to \mathbb{R}$ defined by $\tau_i(x) = x_i$, which naturally extends to sets. Denote the unit sphere $\mathbb{S}^n$ and the unit disk $\mathbb{D}^n$, where $\mathbb{S}^n := \{x \in \mathbb{R}^{n+1} \mid \|x\| = 1\}$ and $\mathbb{D}^{n+1} := \{x \in \mathbb{R}^{n+1} \mid \|x\| \leq 1\}$. For sets $A, B \subseteq \mathbb{R}^n$ we define the binary relation $A \prec B$ ($A \preceq B$) if $\tau_1(A) \cap \tau_1(B)$ is non-empty and for all $x \in A$ and $y \in B$ such that $x_1 = y_1$, we have $x_2 < y_2$ ($x_2 \leq y_2$).

Denote the space of n-times continuously differentiable functions from A into B as $C^n(A,B)$. We use the notation $F : A \rightrightarrows B$ to denote a set-valued map from A into B. For $A \subset X$ and $f : X \to Y$, we define the image of A under f as $f(A) := \{f(x) \in Y \mid x \in A\}$. We denote the space of piecewise continuous signals on A as $S(A) := PC(\mathbb{R}_+, A)$. Denote the unit interval $I := [0,1]$. For the set $A \subset \mathbb{R}^2$, we will call a path $\gamma \in C^0(I,A)$ simple if γ is injective. We will call it closed if $\gamma(0) = \gamma(1)$. We define the Cone at vertex $x \in \mathbb{R}^n$ with respect to $a_1, a_2, \ldots, a_k \in \mathbb{R}^n$ as $\text{Cone}_{\{a_1,a_2,\ldots,a_k\}}(x) := \{y \in \mathbb{R}^n \mid \langle y - x | a_i \rangle \geq 0 \ \forall\ i \in \{1,2,\ldots,k\}\}$. For $x \in \mathbb{R}^2$, we use the shorthand notation $\text{Cone}_+(x) := \text{Cone}_{\{\hat{e}_1,\hat{e}_2\}}(x) \subset \mathbb{R}^2$ and $\text{Cone}_-(x) := \text{Cone}_{\{-\hat{e}_1,-\hat{e}_2\}}(x) \subset \mathbb{R}^2$. We use the following continuity definition for set-valued maps [21].

Definition 1. For metric spaces A and D, a set-valued map $F : A \rightrightarrows D$ is said *upper hemi-continuous* at $x \in A$ if for all $\epsilon > 0$ there is $\eta > 0$ such that $F(y) \subset B(F(x),\epsilon)$ for all $y \in B(x,\eta)$.

We next introduce a set characterization useful in formulating safety control problems for order preserving systems.

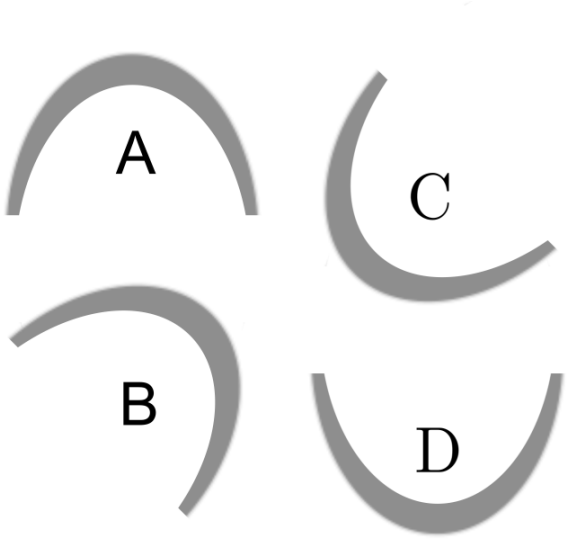


Figure 2.2: The sets $A, B \subset \mathbb{R}^2$ are o.p.c., while the sets $C, D \subset \mathbb{R}^2$ are not o.p.c.

Definition 2. A path $\gamma \in C^0(I, \mathbb{R}^2)$ is said to be *order preserving connected* (o.p.c.) if it is simple, and for all $x \in \mathbb{R}^2$ $\text{Cone}_+(x) \cap \gamma(I) \neq \emptyset$ implies that $\text{Cone}_+(x) \cap \gamma(I)$ is path connected. A set $D \subseteq \mathbb{R}^2$ is said o.p.c. if for all $x, y \in D$, there exists a $\gamma \in C^0(I, D)$ such that $\gamma(0) = x, \gamma(1) = y$ and γ is o.p.c. (Figure 2.2).

Note that any convex set is trivially o.p.c. A partial order is a set P with a partial order relation "$\leq$", which we denote by the pair (P,$\leq$) [32]. In this paper, we are mostly concerned with the partial order $(\mathbb{R}^n, \leq)$ defined by component-wise ordering, that is, for all $w, z \in \mathbb{R}^n$ we have that $w \leq z$ if and only if $w_i \leq z_i$ for all $i \in \{1, 2, \ldots, n\}$. Given sets $A, B \subset \mathbb{R}^n$, we say $A \leq B$ if $a \leq b$ for all $a \in A$ and $b \in B$. For $U \subseteq \mathbb{R}^m$, we define the partial order $(S(U), \leq)$ by component-wise ordering for all time, that is, for all $\mathbf{w}, \mathbf{z} \in S(U)$ we have that $\mathbf{w} \leq \mathbf{z}$ provided $\mathbf{w}(t) \leq \mathbf{z}(t)$ for all $t \in \mathbb{R}_+$. Suppose (P,$\leq_P$) and (Q,$\leq_Q$) are two partially ordered sets. A map $f : P \to Q$ is an order preserving map provided $x \leq_P y$ implies $f(x) \leq_Q f(y)$.

2.3 Class of Systems Considered

We consider piecewise continuous systems, with imperfect state information. This includes the set of hybrid systems with no continuous state reset and no discrete state memory, also referred to as switched systems [28].

Definition 3. A *piecewise continuous system* Σ with imperfect state information is a collection $\Sigma = (X, U, O, f, h)$, in which

(i) $X \subset \mathbb{R}^n$ is a set of continuous variables;

(ii) $U \subset \mathbb{R}^m$ is a set of continuous inputs;

(iii) $O \subset X$ is a set of continuous outputs;

(iv) $f : X \times U \to X$ is a *piecewise continuous* vector field;

(v) $h : O \rightrightarrows X$ is an output map.

For an output measurement $z \in O$, the function $h(z)$ returns the set of all states compatible with the current output. We assume h is closed valued, that is, for all $z \in O$, $h(z)$ is closed. We assume that there is a $\bar{z} \in O$ such that $h(\bar{z}) = X$, corresponding to missing sensory information. We let $\phi(t, x, \mathbf{u})$ denote the flow of Σ at time $t \in \mathbb{R}_+$, with initial condition $x \in X$ and input $\mathbf{u} \in S(U)$ [67]. Denote the i^{th} component of the flow by $\phi_i(t, x, \mathbf{u})$.

We restrict the class of piecewise systems to order preserving systems. These systems are defined on the partial orders $(\mathbb{R}^n, \leq)$ and $(S(U), \leq)$ as follows.

Definition 4. The system $\Sigma = (X, U, O, f, h)$ is an *order preserving* system provided there exist constants $u_L, u_H \in \mathbb{R}^m$ and $\xi > 0$ such that

(i) $U = [u_L, u_H] \subset \mathbb{R}^m$;

(ii) The flow $\phi(t, x, \mathbf{u})$ is an order preserving map with respect x and $\mathbf{u}$;

(iii) $f_1(x, u) \geq \xi$ for all $(x, u) \in X \times U$;

(iv) For all $z \in O$, $h(z) = [\inf h(z), \sup h(z)] \subseteq \mathbb{R}^n$.

Conditions for establishing order preserving properties of the flow generated by a

smooth vector field $f(x,u)$ have been previously addressed [15]. Sufficient conditions for establishing order preserving properties of piecewise-affine systems have been addressed in [19]. For systems in which x_1 is a position (as in the case of the example illustrated in §2.1), condition (iii) guarantees that the system never comes to a halt. More generally, it enforces the liveness of the system. Condition (iv) requires that the set $h(z)$ for any measurement z is an interval in the $(\mathbb{R}^n, \leq)$ partial order. We next define the parallel composition of two systems as defined in standard references [53].

Definition 5. For $\Sigma^1 = (X^1, U^1, O^1, f^1, h^1)$ and $\Sigma^2 = (X^2, U^2, O^2, f^2, h^2)$, we define the *parallel composition* $\Sigma = \Sigma^1 || \Sigma^2 := (X, U, O, f, h)$, in which $X := X^1 \times X^2$, $U := U^1 \times U^2$, $O := O^1 \times O^2$, $f := (f^1, f^2)$ and $h := (h^1, h^2)$.

For $x = (x^1, x^2) \in X^1 \times X^2$ and $\mathbf{u} = (\mathbf{u}^1, \mathbf{u}^2) \in S(U^1 \times U^2)$, we denote the flow of the parallel composition $\Sigma^1 || \Sigma^2$ as $\phi(t, x, \mathbf{u}) = (\phi^1(t, x^1, \mathbf{u}^1), \phi^2(t, x^2, \mathbf{u}^2))$ in which $\phi^1(t, x^1, \mathbf{u}^1) \in X^1$ and $\phi^2(t, x^2, \mathbf{u}^2) \in X^2$. We denote $\phi_j(t, x, \mathbf{u}) := (\phi_j^1(t, x^1, \mathbf{u}^1), \phi_j^2(t, x^2, \mathbf{u}^2))$.

We next define a new partial order $(S(U), \trianglelefteq)$ on input signals of the parallel composition of two systems as follows.

Definition 6. Given the parallel composition $\Sigma = \Sigma^1 || \Sigma^2$, the input set $U = U^1 \times U^2$, and $\mathbf{u}, \mathbf{v} \in S(U)$, we say that $\mathbf{u} \trianglelefteq \mathbf{v}$ if $\mathbf{u}^1 \geq \mathbf{v}^1$ and $\mathbf{u}^2 \leq \mathbf{v}^2$.

Proposition 1. *Consider $\Sigma = \Sigma^1 || \Sigma^2$ in which Σ^1 and Σ^2 are order preserving systems. For $x \in X$ and input signals $\mathbf{u}, \mathbf{v} \in S(U)$ such that $\mathbf{u} \trianglelefteq \mathbf{v}$, we have that $\phi_1(\mathbb{R}_+, x, \mathbf{u}) \leq \phi_1(\mathbb{R}_+, x, \mathbf{v})$.*

The proof is given in the Appendix. This proposition states that if two inputs satisfy the "$\trianglelefteq$" relation, the trajectories generated by these inputs (with the same initial condition) must satisfy the "$\leq$" relation, that is, one trajectory will always "lie above" the other in the (x_1^1, x_1^2) subspace.

2.4 Problem Formulation

In order to formulate the control problem, we first specify what inputs of $\Sigma = \Sigma^1||\Sigma^2$ are controlled and what are uncontrolled (disturbances). This is performed by introducing a two-player game structure on the parallel composition of the two systems as follows.

Definition 7. A *two-player piecewise continuous game structure* is a tuple $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$ in which

(i) $\Sigma = \Sigma^1||\Sigma^2 = (X, U, O, f, h)$ with Σ^1 and Σ^2 piecewise continuous systems;

(ii) $\Omega, \Delta \subset \mathbb{R}^m \times \mathbb{R}^m$ are the control and disturbance sets, respectively;

(iii) $\varphi : \Omega \times \Delta \to U$ is the game input map;

(iv) $\mathbf{B} \subset X$ is a set of bad states.

The disturbance $\delta \in \Delta$ and the control $\omega \in \Omega$ determine the input $u = (u^1, u^2)$ of Σ through the map φ, that is, we have that $u = \varphi(\omega, \delta)$. Extend the map φ to operate on signals by $\varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}) := \mathbf{u}$ where $\mathbf{u}$ is the signal such that $\mathbf{u}(t) = \varphi(\omega(t), \mathbf{d}(t))$. We denote the flow of the game by $\phi(t, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}))$. We will say that the disturbance δ wins the game if the flow of $\mathscr{G}$ enters $\mathbf{B}$, while the controller ω wins the game if the flow of $\mathscr{G}$ never enters $\mathbf{B}$.

Definition 8. A game structure $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$ is an *order preserving* game structure provided

(i) $\Sigma = \Sigma^1||\Sigma^2$ with Σ^1 and Σ^2 order preserving systems;

(ii) $\Delta := [\delta_L^1, \delta_H^1] \times [\delta_L^2, \delta_H^2] := [\delta_L, \delta_H]$ and $\Omega := [\omega_L^1, \omega_H^1] \times [\omega_L^2, \omega_H^2] := [\omega_L, \omega_H]$;

(iii) The game input $\varphi(\omega, \delta) = (\varphi^1(\omega^1, \delta^1), \varphi^2(\omega^2, \delta^2))$ is an order preserving map with respect to control ω and disturbance δ;

(iv) $\mathbf{B} := \{x \in \mathbb{R}^n \times \mathbb{R}^n \mid (x_1^1, x_1^2) \in B\}$ with B an o.p.c. set.

The order preserving property of φ can be interpreted as follows. For the control signals $\omega, \mathbf{w} \in S(\Omega)$ and disturbance signals $\mathbf{d}, \mathbf{d} \in S(\Delta)$, if we have that $\omega \preceq \mathbf{w}$ and $\mathbf{d} \preceq \mathbf{d}$, then $\varphi(\omega, \mathbf{d}) \preceq \varphi(\mathbf{w}, \mathbf{d})$. Similarly, $\omega \trianglelefteq \mathbf{w}$ and $\mathbf{d} \trianglelefteq \mathbf{d}$ implies $\varphi(\omega, \mathbf{d}) \trianglelefteq \varphi(\mathbf{w}, \mathbf{d})$. The utility of this formulation lies in the ability to model cooperation and competition between two agents under a simple unified framework. For a cooperative scenario, in which both systems Σ^1 and Σ^2 are affected by the control but not by the disturbance, we let $\varphi_{\text{coop}}(\omega, \delta) := \omega$. For a competitive scenario, in which system Σ^2 is an adversary while system Σ^1 is completely controlled, we have $\varphi_{\text{comp}}(\omega, \delta) := (\omega^1, \delta^2)$. The more general case, in which both systems Σ^1 and Σ^2 are affected by control and disturbance, could represent model uncertainty for example. An instance of each case is presented in §2.7. One can easily check that the example proposed in §2.1 is an order preserving game structure in which $\varphi = \varphi_{\text{coop}}$.

In the reminder of this paper, we assume (unless stated otherwise) that the flow of $\mathscr{G}$ is continuous with respect to initial condition, with respect to input, and with respect to time. Continuity conditions for the flow of a hybrid system have been previously investigated by, for example, [68] and the references therein. For the compact set of initial conditions $A \subset X$, we assume that the set-valued flow $\phi(t, A, S(U))$ is compact and upper hemi-continuous with respect to time. This property is satisfied, for example, in systems generated by the differential inclusion $\dot{x} \in f(x, U)$, in which $f(x, U)$ is a Marchaud map (see Theorem 3.5.2 in [20], and Corollary 4.5 in [88]). Note that, given a differential inclusion $\dot{x} \in f(x, U)$, the closed convex hull generates a differential inclusion $\dot{x} \in \overline{\text{co}} f(x, U)$, which is Marchaud provided that it is upper hemi-continuous and bounded above by some linear affine function, that is, $\|f(x, U)\| \leq c(\|x\| + 1)$. This allows for the over-approximation of a given system with another one that has the desired properties of the set-valued flow.

Given a game structure $\mathscr{G}$, we consider the problem of designing a controller that on the basis of the output information guarantees that the flow of $\mathscr{G}$ never enters the bad set of states $\mathbf{B}$ for all disturbance choices. For stating the control problem with imperfect state information, denote by $\hat{x}(t, \hat{x}_0, \omega, \mathbf{z})$ the set of all possible states at time t compatible with the set of initial conditions $\hat{x}_0 \subset X$ and measurable signals ω and $\mathbf{z}$. More formally,

$$\hat{x}(t, \hat{x}_0, \omega, \mathbf{z}) := \{x \in X \mid \exists\ x_0 \in \hat{x}_0 \text{ and } \mathbf{d} \in S(\Delta) \text{ s.t. } \phi(t, x_0, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) = x \text{ and } \phi(\tau, x_0, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \in h(\mathbf{z}(\tau))\ \forall\ \tau \in [0, t]\}.$$

The set $\hat{x}(t, \hat{x}_0, \omega, \mathbf{z})$ is called the information state [64] and we will denote it by $\hat{x}(t)$ when $\hat{x}_0,\ \omega$ and $\mathbf{z}$ are clear from the context. We note that if the set of initial conditions $\hat{x}_0$ is compact, then the information state $\hat{x}(t, \hat{x}_0, \omega, \mathbf{z})$ is compact by the compactness of the set-valued flow and the closed value property of the output map $h(z)$.

Problem 1. *(Dynamic Feedback Safety Control Problem)* Given a game structure $\mathscr{G}$, determine the set

$$\bar{\mathcal{W}} := \left\{ A \in 2^X \;\middle|\; \begin{array}{c} \exists\ \omega \in S(\Omega) \text{ s.t. } \forall\ \mathbf{z} \in S(O) \text{ and } \forall\ t \in \mathbb{R}_+ \\ \text{we have } \hat{x}(t, A, \omega, \mathbf{z}) \cap \mathbf{B} = \emptyset \end{array} \right\}$$

and a set-valued map $G : 2^X \rightrightarrows \Omega$ such that for initial convex sets $A \in \bar{\mathcal{W}}$, we have $\hat{x}(t, A, \omega, \mathbf{z}) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$ and $\mathbf{z} \in S(O)$ when $\omega(\tau) \in G(\hat{x}(\tau, A, \omega, \mathbf{z}))$ for all $\tau \in \mathbb{R}_+$.

This problem can be interpreted as one of determining the set of all initial state uncertainties $A \in 2^X$ for which a control map exists, that on the basis of the measurable signals, guarantees that the information state never intersects $\mathbf{B}$.

Problem 2. *(Static Feedback Safety Control Problem)* Given a game structure $\mathscr{G}$ with $O = X$ and h the identity map, determine the set

$$\mathcal{W} := \left\{ x \in X \;\middle|\; \begin{array}{c} \exists\ \omega \in S(\Omega) \text{ s.t. } \forall\ \mathbf{d} \in S(\Delta) \text{ and } \forall\ t \in \mathbb{R}_+ \\ \text{we have } \phi(t, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \notin \mathbf{B} \end{array} \right\}$$

and a set-valued map $g : X \rightrightarrows \Omega$ such that for initial conditions $x \in \mathcal{W}$, we have that $\phi(t, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \notin \mathbf{B}$ for all $\mathbf{d} \in S(\Delta)$ and $t \in \mathbb{R}_+$ when $\omega(\tau) \in g(\phi(\tau, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})))$ for all $\tau \in \mathbb{R}_+$.

This problem can be interpreted as one of determining the set of all initial states $x \in X$ for which a static feedback map exists such that the flow of the system never enters $\mathbf{B}$ for all possible disturbance signals $\mathbf{d}$.

2.5 Problem Solution

In this section, we propose the solution to Problems 1 and 2 by first computing the complement to the sets $\bar{\mathcal{W}}$ and $\mathcal{W}$, then explicitly computing a dynamic and a static feedback map.

2.5.1 Computation of the Sets $\bar{\mathcal{W}}$ and $\mathcal{W}$

Consider $C := X \backslash \mathcal{W}$. This set is named the *capture set* as it represents the set of all initial states for which no matter what control is applied, there is a disturbance that drives the flow into $\mathbf{B}$. It is mathematically represented as

$$C = \{x \in X \mid \forall\, \omega \in S(\Omega), \exists\, \mathbf{d} \in S(\Delta) \text{ and } \ t \in \mathbb{R}_+ \text{ s.t. } \phi(t, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \in \mathbf{B}\}\,.$$

For a fixed control signal $\bar{\omega} \in S(\Omega)$, we define the *restricted capture set* $C_{\bar{\omega}}$ to be the capture set when the control signal is fixed to $\bar{\omega}$. Mathematically, it is expressed as

$$C_{\bar{\omega}} = \{x \in X \mid \exists\, \mathbf{d} \in S(\Delta) \text{ and } \ t \in \mathbb{R}_+ \text{ s.t. } \phi(t, x, \varphi(\bar{\omega}, \mathbf{d})) \in \mathbf{B}\}\,.$$

The restricted capture sets form the basis for our solution to Problems 1 and 2. In the simple example presented in §2.1, two restricted capture sets of relevance, $C_{\omega_{\mathcal{H}}}$ and $C_{\omega_{\mathcal{L}}}$, are represented in Figure 2.1. More generally, for an order preserving game structure

define the constant control $\omega_{\mathcal{L}} := (\omega_H^1, \omega_L^2)$ and $\omega_{\mathcal{H}} := (\omega_L^1, \omega_H^2)$, and corresponding control signals $\omega_{\mathcal{L}}(t) := \omega_{\mathcal{L}}$ and $\omega_{\mathcal{H}}(t) := \omega_{\mathcal{H}}$ for all $t \in \mathbb{R}_+$. For all $\omega \in S(\Omega)$, we have that

$$\omega_{\mathcal{L}} \trianglelefteq \omega \trianglelefteq \omega_{\mathcal{H}}. \tag{2.1}$$

Similarly, define the constant disturbance $\delta_{\mathcal{L}} := (\delta_H^1, \delta_L^2)$ and $\delta_{\mathcal{H}} := (\delta_L^1, \delta_H^2)$, and corresponding disturbance signals $\mathbf{d}_{\mathcal{L}}(t) := \delta_{\mathcal{L}}$ and $\mathbf{d}_{\mathcal{H}}(t) := \delta_{\mathcal{H}}$ for all $t \in \mathbb{R}_+$. For all $\mathbf{d} \in S(\Delta)$, we have that

$$\mathbf{d}_{\mathcal{L}} \trianglelefteq \mathbf{d} \trianglelefteq \mathbf{d}_{\mathcal{H}}. \tag{2.2}$$

We now state the main results of this paper.

Lemma 1. *Consider order preserving game structure $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$ with a convex set $A \subset X$. Let $\omega \in S(\Omega)$ and $\gamma \in C^0(I, \mathbb{R}^2)$ o.p.c. with $\inf \tau_1(A) < \max \tau_1(\gamma(I))$. Then, $\gamma(I) \cap \bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(t, A, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) = \emptyset$ for all $t \in \mathbb{R}_+$ if and only if $\phi_1(\mathbb{R}_+, A, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \succ \gamma(I)$ or $\phi_1(\mathbb{R}_+, A, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \prec \gamma(I)$.*

Before giving the proof, we need the following intermediate results.

Proposition 2. *Consider order preserving game structure $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$ and let $x \in X$, $\omega \in S(U)$, $\mathbf{d} \in S(\Delta)$ and $\gamma \in C^0(I, \mathbb{R}^2)$ o.p.c. where $x_1^1 \leq \max \tau_1(\gamma(I))$. Then, we have that either $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \succ \gamma(I)$ or $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \prec \gamma(I)$ if and only if $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) = \emptyset$.*

Proof. ($\Rightarrow$) Follows from the definition of the $\prec$ relation.

($\Leftarrow$) Suppose $\{\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \succ \gamma(I)$ or $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \prec \gamma(I)\}$ does not hold. The hypothesis $\phi_1^1(0, x^1, \varphi^1(\omega^1, \mathbf{d}^1)) \leq \sup \tau_1(\gamma(I))$ and condition (iii) of Definition 4 imply that there exist $\alpha^1, \alpha^2 \in I$ and $t_1, t_2 \in \mathbb{R}_+$ such that $\phi_1(t_1, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \preceq \gamma(\alpha^1)$ and $\phi_1(t_2, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \succeq \gamma(\alpha^2)$. For simplifying notation, let $\zeta(t) := \phi_1(t, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}))$. Without loss of generality, assume $\alpha^1 \leq \alpha^2$, define $\chi \in \mathbb{R}^2$ where $\chi_1 := \min\left\{\gamma_1(\alpha^1), \gamma_1(\alpha^2)\right\}$ and

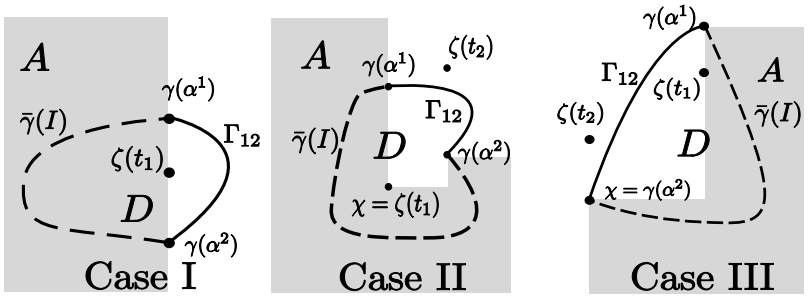


Figure 2.3: Three Cases.

$\chi_2 := \min\{\phi_1^2(t_1, x^2, \varphi^2(\omega^2, \mathbf{d}^2)), \gamma_2(\alpha^2)\}$. Next, we define $\Gamma_{12} := \gamma([\alpha^1, \alpha^2])$. By the construction of χ, we have that $\gamma(\alpha^1), \gamma(\alpha^2) \in \text{Cone}_+(\chi)$, which implies that $\Gamma_{12} \subset \text{Cone}_+(\chi)$ by the definition of o.p.c. We now consider the three possible cases: (Case I) $t_1 = t_2$, (Case II) $t_1 < t_2$, and (Case III) $t_1 > t_2$.

(Case I) Suppose $t_1 = t_2$, implying $\gamma(\alpha^2) \preceq \zeta(t_1) \preceq \gamma(\alpha^1)$. Consider the open half space $A :=\sim \text{Cone}_{\{e_1\}}(\chi) \subset \mathbb{R}^2$ which is trivially path connected, and the set $\tilde{A} := A \cup \gamma(\alpha^1) \cup \gamma(\alpha^2)$. The set $\tilde{A}$ is also path connected, implying the existence of a path $\bar{\gamma} \in C^0(I, \tilde{A})$ such that $\bar{\gamma}(0) = \gamma(\alpha^1)$ and $\bar{\gamma}(1) = \gamma(\alpha^2)$ where $\bar{\gamma}$ is simple. Since $\Gamma_{12} \subset \text{Cone}_+(\chi)$, and $\text{Cone}_{\{-\hat{e}_1\}}(\chi) \cap \text{Cone}_+(\chi) = \emptyset$ by definition of the cone, we must have that $A \cap \Gamma_{12} = \emptyset$. This implies that $\bar{\gamma}(I)$ only intersects Γ_{12} at $\bar{\gamma}(0)$ and $\bar{\gamma}(1)$, allowing us to re-parameterize $\bar{\gamma}(I) \cup \Gamma_{12}$ with a simple closed curve (see Figure 2.3).

This simple closed curve, by the Jordan Curve Theorem [74], partitions $\mathbb{R}^2$ into two sets, D bounded and $\sim D$ unbounded. By construction, D is such that $\zeta(t_1) \in D$ and $\partial D = \Gamma_{12} \cup \bar{\gamma}(I)$. Condition (iii) of Definition 4 implies that $\|\zeta(t)\| \to \infty$ as $t \to \infty$. Thus, $\zeta([\bar{t}, \infty)) \cap \partial D$ must be non-empty because D is a bounded set. Since condition (iii) of Definition 4 implies that $\zeta([\bar{t}, \infty)) \cap \tilde{A}$ is empty and $\bar{\gamma}(I) \subset \tilde{A}$, we must have that $\zeta([\bar{t}, \infty)) \cap \Gamma_{12} \neq \emptyset$. This in turn implies $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) \neq \emptyset$.

(Case II) Suppose $t_1 < t_2$. This along with condition (ii) of Definition 4 implies that $\gamma_1(\alpha^1) < \gamma_1(\alpha^2)$. We assume that $\zeta(t_1) \preceq \gamma(I)$ and $\zeta(t_2) \succeq \gamma(I)$, otherwise we would be

back to (Case I). Define the sets $S_1 := \text{Cone}_{\{\hat{e}_1,-\hat{e}_2\}}(\gamma(\alpha^2))$ and $S_2 := \text{Cone}_+(\chi)$. Define $A := \mathring{S}_1 \cup (\sim S_2)$ and $\tilde{A} := A \cup \gamma(\alpha^1) \cup \gamma(\alpha^2)$. Since γ is an o.p.c. path, $\Gamma_{12} \subset \text{Cone}_+(\chi)$ and $\Gamma_{12} \cap S_1 = \emptyset$, we must have that $\Gamma_{12} \cap A = \emptyset$. The set $\tilde{A}$ is path connected, implying the existence of $\bar{\gamma} \in C^0(I, \tilde{A})$ with $\bar{\gamma}(0) = \gamma(\alpha^1)$, $\bar{\gamma}(1) = \gamma(\alpha^2)$ and $\bar{\gamma}$ simple. Since $A \cap \Gamma_{12} = \emptyset$, $\bar{\gamma}(I) \cup \Gamma_{12}$ can be re-parameterized with a simple closed curve (see Figure 2.3). This curve, by the Jordan Curve Theorem, forms a bounded set D, where $\zeta(t_1) \in D$ by construction. Condition (ii) and (iii) of Definition 4 along with the decoupling of the dynamics imply that $\zeta([t_1, \infty]) \cap A = \emptyset$ and $\zeta([t_1, \infty]) \cap \partial D \neq \emptyset$. Since $\bar{\gamma} \subset A$, we have that $\zeta([t_1, \infty]) \cap \Gamma_{12} \neq \emptyset$. Therefore, $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) \neq \emptyset$.

(Case III) Suppose $t_2 < t_1$, which along with condition (iii) of Definition 4 implies that $\gamma_1(\alpha^2) < \gamma_1(\alpha^1)$. We assume that $\zeta(t_1) \preceq \gamma(I)$ and $\zeta(t_2) \succeq \gamma(I)$, otherwise we would be back to (Case I). Define the sets $P := \text{Cone}_{\{e_1\}}(\chi)$, $R := \text{Cone}_{\{e_1,-e_2\}}(\gamma(\alpha^1))$, $H := \text{Cone}_{(+)}(\chi) \backslash \mathring{R}$, $A := P \backslash H$, and $\tilde{A} := A \cup \gamma(\alpha^1) \cup \gamma(\alpha^2)$. The set $\tilde{A}$ is path connected, implying the existence of $\bar{\gamma}$ where $\bar{\gamma} \in C^0(I, \tilde{A})$ with $\bar{\gamma}(0) = \gamma(\alpha^1), \bar{\gamma}(1) = \gamma(\alpha^2)$ and $\bar{\gamma}$ simple. Observe that $A \cap \Gamma_{12} = \emptyset$, thus $\bar{\gamma}(I) \cup \Gamma_{12}$ can be re-parametrized with a simple closed curve. We invoke the Jordan Curve Theorem to construct the bounded set D, where $\zeta(t_1) \in D$ by construction (Figure 2.3). By construction, we also have that $\zeta(t_2) \notin D$. Thus, the uniform continuity of the flow with respect to time implies $\zeta([t_2, t_1]) \cap \partial D \neq \emptyset$. Condition (iii) of Definition 4 implies that $\zeta([t_2, t_1]) \subset H$, thus implying $\zeta([t_2, t_1]) \cap \tilde{A} = \emptyset$. Since $\partial D = \Gamma_{12} \cup \bar{\gamma}(I)$ and $\bar{\gamma}(I) \subset \tilde{A}$, we must have $\zeta([t_2, t_1]) \cap \Gamma_{12} \neq \emptyset$. This implies that $\zeta([t_2, t_1]) \cap \gamma(I) \neq \emptyset$, giving the desired result $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) \neq \emptyset$.

Therefore, we have shown for each case $\phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) \neq \emptyset$, completing the proof. □

Proposition 2 states that the flow ϕ generated from the initial condition x, controlled input ω and disturbance $\mathbf{d}$ can avoid an o.p.c. path γ in the (x_1^1, x_1^2) subspace if and only

if the trajectory of ϕ_1 lies above $\gamma(I)$ or if the trajectory of ϕ_1 lies below $\gamma(I)$. Another intermediate result is needed before stating the proof of Lemma 1.

Proposition 3. *Consider order preserving game structure* $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$*,* $x \in X$*,* $\omega \in S(U)$ *and* $\gamma \in C^0(I, \mathbb{R}^2)$ *o.p.c. with* $x_1^1 \leq \max \tau_1(\gamma(I))$*. If* $\bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) = \emptyset$*, then either* $\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \succ \gamma(I)$ *or* $\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \prec \gamma(I)$*.*

Proof. The assumption that $\gamma(I) \cap \bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) = \emptyset$ implies (a) $\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \cap \gamma(I) = \emptyset$ and (b) $\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \cap \gamma(I) = \emptyset$. From Proposition 2, we have that (a) implies either

$$\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \succ \gamma(I) \tag{2.3}$$

$$\text{or} \qquad \phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \prec \gamma(I). \tag{2.4}$$

Similarly, Proposition 2 along with (b) implies either

$$\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \succ \gamma(I) \tag{2.5}$$

$$\text{or} \qquad \phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \prec \gamma(I). \tag{2.6}$$

If (2.3) is satisfied, we immediately obtain the result. Similarly, if (2.4) and (2.6) are satisfied the result also follows. Therefore, we are left with showing that relations (2.4) and (2.5) are not both possible. By contradiction, assume they are both possible and define the constant signals $\mathbf{d}_L^2(t) := \delta_L^2$, $\mathbf{d}_H^2(t) := \delta_H^2$, $\mathbf{d}_L^1(t) := \delta_L^1$, and $\mathbf{d}_H^1(t) := \delta_H^1$ for all $t \in \mathbb{R}_+$. Then, there is $(\alpha^1, \alpha^2) \in \gamma(I)$, $t_a > 0$, and $t_b > 0$ such that $\phi_1^2(t_a, x^2, \varphi^2(\omega^2, \mathbf{d}_L^2)) < \alpha^2$ and $\phi_1^2(t_b, x^2, \varphi^2(\omega^2, \mathbf{d}_H^2)) > \alpha^2$. Since $\phi_1^1(t_a, x^1, \varphi^1(\omega^1, \mathbf{d}_H^1)) = \alpha^1 = \phi_1^1(t_b, x^1, \varphi^1(\omega^1, \mathbf{d}_L^1))$, the order preserving property of φ in its arguments imply that $t_a \leq t_b$. For fixed x^2 and ω^2, define the function $\Phi_1^2 : [t_a, t_b] \times S(\Delta^2) \to \mathbb{R}$ by $\Phi_1^2(t, \mathbf{d}^2) := \phi_1^2(t, x^2, \varphi^2(\omega^2, \mathbf{d}^2))$. This is a continuous function from a connected metric space into the reals. Therefore, we can apply the intermediate value theorem to state that there is a pair $\bar{t} \in [t_a, t_b]$ and $\bar{\mathbf{d}}^2 \in S(\Delta^2)$ such that $\Phi_1^2(\bar{t}, \bar{\mathbf{d}}^2) = \alpha^2$.

Property (iii) of Definition 4 further implies that the ordering $\phi_1^1(\bar{t}, x^1, \varphi^1(\omega^1, \mathbf{d}_H^1)) > \alpha^1$ and ordering $\phi_1^1(\bar{t}, x^1, \varphi^1(\omega^1, \mathbf{d}_L^1)) < \alpha^1$ must hold. For fixed x^1 and ω^1, define the map $\Phi_1^1 : S(\Delta^1) \to \mathbb{R}$ by $\Phi_1^1(\mathbf{d}^1) := \phi_1^1(\bar{t}, x^1, \varphi^1(\omega^1, \mathbf{d}^1))$. This is a continuous function from a connected metric space to the reals, therefore we can apply again the intermediate value theorem to conclude that there is $\bar{\mathbf{d}}^1 \in S(\Delta^1)$ such that $\Phi_1^1(\bar{\mathbf{d}}^1) = \alpha^1$.

As a consequence, we have that $\phi_1(\bar{t}, x, \varphi(\omega, (\bar{\mathbf{d}}^1, \bar{\mathbf{d}}^2))) = (\alpha^1, \alpha^2) \in \gamma(I)$ for $(\mathbf{d}^1, \mathbf{d}^2) \in S(\Delta)$. This in turn contradicts the assumption that $\bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(\mathbb{R}_+, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) = \emptyset$. □

Proposition 3 states that the flow ϕ generated from the initial condition x and controlled input ω will avoid an o.p.c. path γ in the (x_1^1, x_1^2) subspace if and only if the trajectory of ϕ_1 generated with the disturbance signal $\mathbf{d}_{\mathcal{L}}$ lies above $\gamma(I)$ or if the trajectory of ϕ_1 generated with the disturbance signal $\mathbf{d}_{\mathcal{H}}$ lies below $\gamma(I)$.

Proof. (*Lemma 1*) ($\Leftarrow$) For every disturbance $\mathbf{d} \in S(\Delta)$, we have that $\mathbf{d}_{\mathcal{L}} \trianglelefteq \mathbf{d} \trianglelefteq \mathbf{d}_{\mathcal{H}}$. From Proposition 1, it follows that for every $x \in A$ and $t \in \mathbb{R}_+$, we have that $\phi(t, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \lesssim \phi(t, x, \varphi(\omega, \mathbf{d})) \lesssim \phi(t, x, \varphi(\omega, \mathbf{d}_{\mathcal{H}}))$. Therefore, the result follows directly from the assumption.

($\Rightarrow$) Suppose $\{\phi_1(\mathbb{R}_+, A, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \succ \gamma(I)$ or $\phi_1(\mathbb{R}_+, A, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \prec \gamma(I)\}$ does not hold. Then there must exist $x, y \in A$, $\alpha^1, \alpha^2 \in I$, and $t_1, t_2 > 0$ such that $\phi_1(t_1, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \prec \gamma(\alpha^1)$ and $\phi_1(t_2, y, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \succ \gamma(\alpha^2)$ (the relation is strict, otherwise the result is immediate). We assume that $\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \prec \gamma(I)$, otherwise Proposition 2 implies that $\phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \cap \gamma(I) \neq \emptyset$. Likewise, Proposition 2 implies we must have $\phi_1(\mathbb{R}_+, y, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \succ \gamma(I)$. Furthermore, unless $\phi_1(\mathbb{R}_+, y, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \succ \gamma(I)$ is satisfied, the previous statement along with Proposition 3 implies that $\phi_1(\mathbb{R}_+, y, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \cap \gamma(I) \neq \emptyset$. Figure 2.4 shows the resulting geometry of the flow. Let $\bar{\alpha} \in I$ be such that $\tau_1(\gamma(I)) \leq$

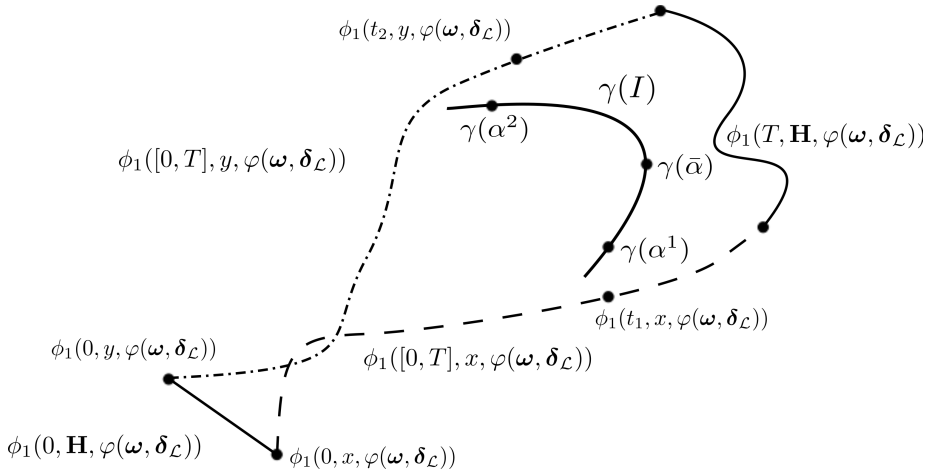


Figure 2.4: Geometry of $\phi_1(t, x, \varphi(\omega, \mathbf{d}_{\mathcal{L}}))$ and $\phi_1(t, y, \varphi(\omega, \mathbf{d}_{\mathcal{L}}))$.

$\tau_1(\gamma(\bar{\alpha}))$. Condition (iii) of Definition 4 leads to $x_1^1 < \phi_1^1(t_1, x^1, \varphi^1(\omega, \mathbf{d}_{\mathcal{L}})) \leq \gamma_1(\bar{\alpha})$ and $y_1^1 < \phi_1^1(t_2, y^1, \varphi^1(\omega, \mathbf{d}_{\mathcal{L}})) \leq \gamma_1(\bar{\alpha})$. Consider $H := \overline{\text{co}}\{x, y\} \subset A$, since convexity is preserved under projection [27], condition (iii) of Definition 4 implies there is $T > 0$ such that

$$\phi_1^1(0, \tau_1(H), \varphi^1(\omega^1, \mathbf{d}_{\mathcal{L}}^1)) < \gamma_1(\bar{\alpha}) < \phi_1^1(T, \tau_1(H), \varphi^1(\omega^1, \mathbf{d}_{\mathcal{L}}^1)). \tag{2.7}$$

We seek to show that $\gamma(\bar{\alpha}) \in \phi_1([0, T], H, \varphi(\omega, \mathbf{d}_{\mathcal{L}}))$. Define $K := [0, T] \times H \subset \mathbb{R}_+ \times \mathbb{R}^{2n}$ and let $\Theta : K \to \mathbb{R}^2$ be the map defined by $\Theta(t, z) := \phi_1(t, z, \varphi(\omega, \mathbf{d}_{\mathcal{L}}))$ for $(t, z) \in K$. We proceed by breaking this proof into three steps:

(i) Construct from Θ a map $\psi : \mathbb{S}^1 \to \mathbb{S}^1$;

(ii) Show that the degree of ψ is nonzero;

(iii) Show that the degree of ψ being nonzero implies that $\gamma(\bar{\alpha}) \in \Theta(K)$.

(i) Denote the four corners of ∂K : $h_1 = (0, x),\ h_2 = (T, x),\ h_3 = (T, y),\ h_4 = (0, y)$. Define the sets $A_1 := \overline{\text{co}}(\{h_1, h_2\}) \cup \text{co}(\{h_2, h_3\})$ and $A_2 := \overline{\text{co}}(\{h_3, h_4\}) \cup \overline{\text{co}}(\{h_4, h_1\})$. Consider the standard covering map of $\mathbb{S}^1$ $p : \mathbb{R} \to \mathbb{S}^1$, in which $p(z) := (cos(2\pi z), sin(2\pi z))$. Define

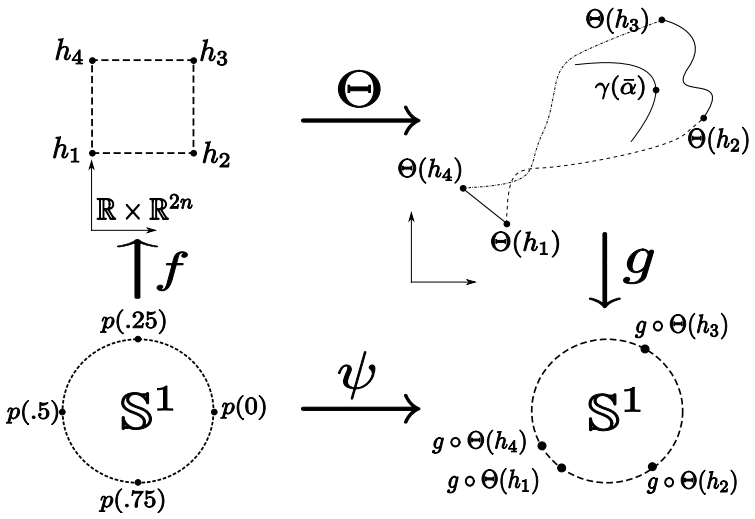
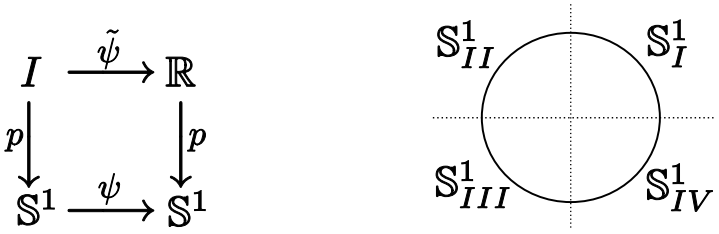


Figure 2.5: The Mapping ψ.



(a) Commuting Diagram (b) Four Quadrants of $\mathbb{S}^1$

Figure 2.6: Tools used to find deg ψ.

the homeomorphism $f : \mathbb{D}^1 \to K$, such that $f(p(0)) = h_1,\ f(p(.25)) = h_2,\ f(p(.5)) = h_3$, and $f(p(.75)) = h_4$. Since Θ is a continuous function, we have that $\Theta(\partial K)$ defines a closed curve. Assume that $\gamma(\bar{\alpha}) \notin \Theta(\partial K)$ and let $g : \mathbb{R}^2 \backslash \gamma(\bar{\alpha}) \to \mathbb{S}^1$ be the continuous map defined by

$$g(z) := \frac{z - \gamma(\bar{\alpha})}{\|z - \gamma(\bar{\alpha})\|},\ \forall z \in \mathbb{R}^2 \backslash \gamma(\bar{\alpha}). \tag{2.8}$$

Define $\psi \in C^0(\mathbb{S}^1, \mathbb{S}^1)$ as $\psi(x) := g \circ \Phi \circ f(x)$ for all $x \in \mathbb{S}^1$ (see Figure 2.5).

(ii) To compute the degree of ψ, we consider the lift $\tilde{\psi} : I \to \mathbb{R}$ where $p \circ \tilde{\psi} = \psi \circ p$ (see Figure 2.6(a)). The degree of ψ is defined as $\deg\psi := \tilde{\psi}(1) - \tilde{\psi}(0)$ (see [65] for details). We introduce the sets $\mathbb{S}^1_I := p([0, .25]), \mathbb{S}^1_{II} := p([.25, .5]), \mathbb{S}^1_{III} := p([.5, .75]), \mathbb{S}^1_{IV} := p([.75, 1])$ (see Figure 2.6(b)). Let $\kappa_1 := \tilde{\psi}(0)$ and note that $p(\kappa_1) = \psi(p(0)) = g(\Theta(h_1))$, which must

be in $\mathbb{S}^1_{III}$, since $\Theta(h_1) < \gamma(\bar{\alpha})$. Let $\kappa_2 = \tilde{\psi}(.5)$ and note that $p(\kappa_2) = \psi(p(.5)) = g(\Theta(h_3))$. From (2.7) and condition (iii) of Definition 4, we have that $\gamma(\bar{\alpha}) < \Theta(h_3)$. This inequality along with the definition of g imply that $g(\Theta(h_3)) \in \mathbb{S}^1_I$. As a consequence, we have $p(\kappa_2) \in \mathbb{S}^1_I$, implying that $\kappa_1 \neq \kappa_2$.

We next show that $\kappa_2 > \kappa_1$. Since $\Theta \circ f(p([0, .5])) = \Theta(A_1)$, equation (2.7) along with condition (iii) of Definition 4 implies that $\Theta(A_1) \prec \gamma(\bar{\alpha})$. This implies that $\psi(p([0, .5])) = g(\Theta(A_1)) \subset \mathbb{S}^1_I \cup \mathbb{S}^1_{IV} \cup \mathbb{S}^1_{III}$. Therefore, if $\psi(p(\zeta))$ cannot enter $\mathbb{S}^1_{II}$ for all $\zeta \in [0, .5]$, then $\kappa_1 < \kappa_2$ by the definition of p.

Finally, let $\kappa_3 := \tilde{\psi}(1)$. We show that $\kappa_2 < \kappa_3$. Since $\Theta \circ f(p([.5, 1])) = \Theta(A_2)$, from (2.7) and Condition (ii) of Definition 4 we have that $\Theta(A_2) \succ \gamma(\bar{\alpha})$. This, along with Condition (iii) of Definition (4) implies that $\psi(p([.5, 1])) = g(\Theta(A_2)) \subset \mathbb{S}^1_I \cup \mathbb{S}^1_{II} \cup \mathbb{S}^1_{III}$. Therefore, if $\psi(p(\zeta))$ cannot enter S^1_{IV} for all $\zeta \in [.5, 1]$, then $\kappa_2 < \kappa_3$ from the definition of p.

We have shown that $\kappa_1 < \kappa_2 < \kappa_3$. As a consequence, $\deg \psi = \tilde{\psi}(1) - \tilde{\psi}(0) = \kappa_3 - \kappa_1 \neq 0$.

(iii) Now suppose we extend the map ψ to $\bar{\psi} \in C^0(\mathbb{D}^1, \mathbb{S}^1)$, where $\bar{\psi}(x) := g \circ \Theta \circ f(x)$ for all $x \in \mathbb{D}^1$. By Lemma 3.5.7 in [65], if a continuous function $h : \mathbb{S}^1 \to \mathbb{S}^1$ extends to a continuous function $H : \mathbb{D}^1 \to \mathbb{S}^1$, then $\deg h$ must be zero. However, we found the degree of ψ to be non-zero, implying that ψ cannot extend to $\bar{\psi}$. Since $\Theta(f(\mathbb{D}^1))$ is well defined, we must have that $g(\Theta(f(\mathbb{D}^1)))$ is undefined. Since $g(z)$ is defined for all $z \in \mathbb{R}^2 \backslash \gamma(\bar{\alpha})$, we must have that $\gamma(\bar{\alpha}) \in \Theta(f(\mathbb{D}))$. This implies that $\gamma(\bar{\alpha}) \in \Theta(K) = \phi_1([0, T], H, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \subset \bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(\mathbb{R}_+, A, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}))$. Therefore, $\bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(\mathbb{R}_+, A, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \gamma(I) \neq \emptyset$. □

Lemma 1 states that the flow in the (x^1_1, x^2_1) subspace generated from the convex set of initial conditions A and control ω can avoid an o.p.c. path $\gamma(I)$ for all disturbance signals if and only if the disturbance signal $\mathbf{d}_{\mathcal{L}}$ takes the trajectory of ϕ_1 above $\gamma(I)$ or the disturbance $\mathbf{d}_{\mathcal{H}}$ takes the trajectory of ϕ_1 below $\gamma(I)$. This result can be generalized to connected sets $A \subset X$ such that $\tau_{1,2}(A)$ is convex, that is, to cases where *only* the projection of the set A

onto the subspace X_1 need be convex.

Theorem 1. *Consider the order preserving game structure* $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$ *with a convex set* $A \subset X$. *Then, the following statements are equivalent*

(*i*) $A \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$ *and* $A \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$;

(*ii*) *For all* $\omega \in S(\Omega)$, *there exist* $\mathbf{d} \in S(\Delta)$ *and* $t \in \mathbb{R}_+$ *such that*

$$\phi(t, A, \varphi(\lambda, \mathbf{u}, \mathbf{d})) \cap \mathbf{B} \neq \emptyset.$$

Proof. ($\Leftarrow$ Contrapositive) By the definition of the restricted capture set, we have that if $A \cap C_{\omega_{\mathcal{L}}} = \emptyset$ then $\phi(t, A, \varphi(\omega_{\mathcal{L}}, \mathbf{d})) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$ and $\mathbf{d} \in S(\Delta)$. Similarly, if $A \cap C_{\omega_{\mathcal{H}}} = \emptyset$ then $\phi(t, A, \varphi(\omega_{\mathcal{H}}, \mathbf{d})) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$ and $\mathbf{d} \in S(\Delta)$.

($\Rightarrow$ Construction) Consider an arbitrary $\omega \in S(\Omega)$. Since $A \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$ and $A \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$, the definition of the restricted capture set implies that there are $x, y \in A, \mathbf{d}_1, \mathbf{d}_2 \in S(\Delta)$ and $t_1, t_2 \in \mathbb{R}_+$ such that $\phi(t_1, x, \varphi(\omega_{\mathcal{L}}, \mathbf{d}_1)) \in \mathbf{B}$ and $\phi(t_2, y, \varphi(\omega_{\mathcal{H}}, \mathbf{d}_2)) \in \mathbf{B}$. Let $\nu, \kappa \in \mathbb{R}^2$ be such that $\nu = \phi_1(t_1, x, \varphi(\omega_{\mathcal{L}}, \mathbf{d}_1))$ and $\kappa = \phi_1(t_2, y, \varphi(\omega_{\mathcal{H}}, \mathbf{d}_2))$. Since $\kappa, \nu \in B$ and B is an o.p.c. set, there exists an o.p.c. path $\gamma \in C^0(I, B)$ with $\gamma(0) = \kappa$ and $\gamma(1) = \nu$.

From equations (2.1)-(2.2) and the order preserving property of φ with respect to control ω and disturbance $\mathbf{d}$, we have that $\varphi(\omega_{\mathcal{L}}, \mathbf{d}_1) \trianglelefteq \varphi(\omega, \mathbf{d}_{\mathcal{H}})$. From Proposition 1, we have that $\phi_1(\mathbb{R}_+, x, \varphi(\omega_{\mathcal{L}}, \mathbf{d}_1)) \preceq \phi_1(\mathbb{R}_+, x, \varphi(\omega, \mathbf{d}_{\mathcal{H}}))$. Since $\phi_1(t_1, x, \varphi(\omega_{\mathcal{L}}, \mathbf{d}_1)) = \nu \in \gamma(I)$ and $x \in A$, this in turn implies that

$$\phi_1(\mathbb{R}_+, A, \varphi(\omega, \mathbf{d}_{\mathcal{H}})) \not\prec \gamma(I). \tag{2.9}$$

From equations (2.1)-(2.2) and the order preserving property of φ with respect to control ω and disturbance $\mathbf{d}$, we have that $\varphi(\omega, \mathbf{d}_{\mathcal{L}}) \trianglelefteq \varphi(\omega_{\mathcal{H}}, \mathbf{d}_2)$. From Proposition 1, we have that $\phi_1(\mathbb{R}_+, y, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \preceq \phi_1(\mathbb{R}_+, y, \varphi(\omega_{\mathcal{H}}, \mathbf{d}_2))$. Since also $\phi_1(t_2, y, \varphi(\omega_{\mathcal{H}}, \mathbf{d}_2)) = \kappa \in \gamma(I)$ and $y \in A$, we have that

$$\phi_1(\mathbb{R}_+, A, \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \not\succ \gamma(I). \tag{2.10}$$

Note that $y_1 < \kappa_1$ from condition (iii) of Definition 4, implying that $\inf \tau_1(A) < \max \tau_1(\gamma(I))$. Therefore, equations (2.9)-(2.10) and Lemma 1 imply that $\gamma(I) \cap \bigcup_{\mathbf{d} \in S(\Delta)} \phi_1(t, A, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \neq \emptyset$ for some $t \in \mathbb{R}_+$. This in turn implies, since $\gamma(I) \subset B$, that there are $\bar{\mathbf{d}} \in S(\Delta)$ and $t \in \mathbb{R}_+$ such that $\phi_1(t, A, \varphi(\omega, \bar{\mathbf{d}})) \cap B \neq \emptyset$. This leads to $\phi(t, A, \varphi(\omega, \bar{\mathbf{d}})) \cap \mathbf{B} \neq \emptyset$. Since this holds for arbitrary $\omega \in S(\Omega)$, we have completed the proof. □

Corollary 1. *For an order preserving game structure $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$, we have that $C = C_{\omega_{\mathcal{H}}} \cap C_{\omega_{\mathcal{L}}}$.*

Proof. ($\subset$) This follows from the definition of C. ($\supset$) Suppose we have that the initial condition $x \in C_{\omega_{\mathcal{H}}} \cap C_{\omega_{\mathcal{L}}}$. Consider any input signal $\omega \in S(\Omega)$. Since $\tau_{1,2}(\{x\})$ is trivially convex, by Theorem 1 there are $\mathbf{d} \in S(\Delta)$ and $t \in \mathbb{R}_+$ such that $\phi(t, \{x\}, \varphi(\omega, \mathbf{d})) \cap \mathbf{B} \neq \emptyset$, implying $x \in C$. □

Theorem 1 states that an initial convex state uncertainty is taken to intersect **B** independently of the control input if and only if it intersects both restricted capture sets $C_{\omega_{\mathcal{H}}}$ and $C_{\omega_{\mathcal{L}}}$. By the corollary, a known initial state is taken to **B** independently of the control input if and only if it is in both $C_{\omega_{\mathcal{H}}}$ and $C_{\omega_{\mathcal{L}}}$.

2.5.2 The Control Map

For an order preserving game structure $\mathscr{G}$, if an initial convex state uncertainty A does not intersect both $C_{\omega_{\mathcal{H}}}$ and $C_{\omega_{\mathcal{L}}}$, from Theorem 1 a control ω exists such that $\phi(t, A, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}))$ never intersects **B** for all **d**. Since $\hat{x}(t, A, \omega, \mathbf{z}) \subseteq \bigcup_{\mathbf{d} \in S(\Delta)} \phi(t, A, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}))$, there must also exist a control ω such that $\hat{x}(t, A, \omega, \mathbf{z})$ never intersects **B**. We thus construct such a control as a feedback map from the current state uncertainty $\hat{x}$. For this purpose, define for an

element $Z \in 2^X$, the set-valued map $G : 2^X \rightrightarrows \Omega$ as

$$G(Z) := \begin{cases} \omega_{\mathcal{L}} & \text{if } Z \cap C_{\omega_{\mathcal{H}}} \neq \emptyset \text{ and } Z \cap \partial C_{\omega_{\mathcal{L}}} \neq \emptyset \text{ and } Z \cap C_{\omega_{\mathcal{L}}} = \emptyset \\ \omega_{\mathcal{H}} & \text{if } Z \cap C_{\omega_{\mathcal{L}}} \neq \emptyset \text{ and } Z \cap \partial C_{\omega_{\mathcal{H}}} \neq \emptyset \text{ and } Z \cap C_{\omega_{\mathcal{H}}} = \emptyset \\ \omega_{\mathcal{L}} & \text{if } Z \cap \partial C_{\omega_{\mathcal{H}}} \neq \emptyset \text{ and } Z \cap \partial C_{\omega_{\mathcal{L}}} \neq \emptyset \\ & \quad \text{and } Z \cap (C_{\omega_{\mathcal{H}}} \cup C_{\omega_{\mathcal{L}}}) = \emptyset \\ \Omega & \text{otherwise.} \end{cases} \tag{2.11}$$

We call the pair $(\mathcal{G}, G)$ a *control system*, where given the initial conditions $A \subset X$ and measurement $\mathbf{z} \in S(O)$, the control system $(\mathcal{G}, G)$ *generates* the *feedback* $\omega^{cl} \in S(\Omega)$ and the *closed-loop* information state $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z})$. The feedback must satisfy the set-valued map G for all time, namely $\omega^{cl}(t) \in G(\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}))$ for all $t \in \mathbb{R}$.

We next show that the control system $(\mathcal{G}, G)$, where $\mathcal{G}$ is an order preserving game structure and G is given by (2.11), generates a closed-loop information state that never intersects $\mathbf{B}$ provided the initial conditions $A \subset X$ are compact, connected, and $A \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $A \cap C_{\omega_{\mathcal{L}}} = \emptyset$.

Theorem 2. *Let $\mathcal{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$ be an order preserving game structure, $(\mathcal{G}, G)$ be the control system generated by the static set-valued feedback (2.11), and let $A \subset X$ be compact and convex. If $A \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $A \cap C_{\omega_{\mathcal{L}}} = \emptyset$, then for arbitrary $\mathbf{z} \in S(O)$ we have that $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$ under $(\mathcal{G}, G)$.*

Proof. First, note that if $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_\omega = \emptyset$ for some $\omega \in S(\Omega)$, then necessarily $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap \mathbf{B} = \emptyset$ because $\mathbf{B} \subset C_\omega$. Thus, we show that if $A \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $A \cap C_{\omega_{\mathcal{L}}} = \emptyset$, then $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$ for all $t \in \mathbb{R}_+$.

We proceed by constructing a modified control system $(\mathcal{G}, \hat{G})$ with a dynamic set-valued map $\hat{G}$, that differs from G only if the argument $Z \subset X$ is such that $Z \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$ and $Z \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$. Denote the closed-loop information state generated by the modified control system as $\hat{y}^{cl}(t, A, \omega^{cl}, \mathbf{z})$. We will show that $\hat{y}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$ or

$\hat{y}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} = \emptyset$ for all $t \in \mathbb{R}_+$. We then show that this implies that the feedback generated by the modified control system $(\mathscr{G}, \hat{G})$ is no different from the feedback generated by the original control system $(\mathscr{G}, G)$. Thus, we also have that $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$ or $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} = \emptyset$ for all $t \in \mathbb{R}_+$.

We now define the dynamic set-valued feedback $\hat{G} : \mathbb{R}_+ \times S(2^X) \rightrightarrows \Omega$ as follows. For the time varying set $\mathbf{Z} \subset S(2^X)$ and time $t \in \mathbb{R}_+$, we define $\hat{G}(t, \mathbf{Z})$ as

$$\hat{G}(t, \mathbf{Z}) := \begin{cases} G(\mathbf{Z}(t)) & \text{if } \mathbf{Z}(t) \cap C_{\omega_{\mathcal{L}}} = \emptyset \text{ or } \mathbf{Z}(t) \cap C_{\omega_{\mathcal{H}}} = \emptyset \\ G(\mathbf{Z}(t^*)) & \text{else, where } t^* := \sup\{\zeta \in [0, t] \mid \mathbf{Z}(\zeta) \cap C_{\omega_{\mathcal{L}}} = \emptyset \\ & \qquad \text{or } \mathbf{Z}(\zeta) \cap C_{\omega_{\mathcal{H}}} = \emptyset\} \end{cases} \tag{2.12}$$

We will now show that the closed-loop information state $\hat{y}^{cl}(t, A, \omega^{cl}, \mathbf{z})$ generated by the control system $(\mathscr{G}, \hat{G})$ never intersects both $C_{\omega_{\mathcal{H}}}$ and $C_{\omega_{\mathcal{L}}}$ at a single time $t \in \mathbb{R}$.

We proceed by contradiction. Suppose that given the measurement $\mathbf{z} \in S(O)$, there exists a time $t_1 > 0$ and feedback $\bar{\omega}^{cl} \in S(\Omega)$ generated by $(\mathscr{G}, \hat{G})$ such that $\hat{y}^{cl}(t_1, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$ and $\hat{y}^{cl}(t_1, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$. Define the times

$$t_{\mathcal{L}} := \inf\{t \in [0, t_1] \mid \hat{y}^{cl}(\zeta, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} \neq \emptyset \ \forall\ \zeta \in [t, t_1]\} \tag{2.13}$$

$$t_{\mathcal{H}} := \inf\{t \in [0, t_1] \mid \hat{y}^{cl}(\zeta, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} \neq \emptyset \ \forall\ \zeta \in [t, t_1]\}. \tag{2.14}$$

Let the maximum of these two times be $\bar{t} := \max\{t_{\mathcal{L}},\ t_{\mathcal{H}}\}$. We must have one of the following cases: (I) $t_{\mathcal{L}} > t_{\mathcal{H}}$; (II) $t_{\mathcal{L}} < t_{\mathcal{H}}$; (III) $t_{\mathcal{L}} = t_{\mathcal{H}}$.

Case(I). From definition (2.14), $t_{\mathcal{H}} < \bar{t}$ implies that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$. We first show that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$.

Suppose that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$. By the definition of the closed-loop information state, there exists $x_0 \in A$ and a disturbance $\mathbf{d} \in S(\Delta)$ such that $\phi(\bar{t}, x_0, \varphi(\bar{\omega}^{cl}, \mathbf{d})) \in C_{\omega_{\mathcal{L}}}$ and $\phi(\tau, x_0, \varphi(\bar{\omega}^{cl}, \mathbf{d})) \in \hat{y}^{cl}(\tau, A, \bar{\omega}^{cl}, \mathbf{z})$ for all $\tau \in [0, \bar{t}]$. For notation, let $\nu := \phi(\bar{t}, x_0, \varphi(\bar{\omega}^{cl}, \mathbf{d}))$. Since the flow is continuous with respect to initial conditions, one can show that $\mathbf{B}$ open

implies that $C_{\omega_{\mathcal{L}}}$ is open. Therefore, we can find $\epsilon > 0$ such that $B(\nu, \epsilon) \subset C_{\omega_{\mathcal{L}}}$. By the continuity of the flow with respect to time, we can find $\eta > 0$ such that if $t \in]\bar{t} - \eta, \bar{t}]$, then $\phi(t, x_0, \varphi(\bar{\omega}^{cl}, \mathbf{d})) \in B(\nu, \epsilon) \subset C_{\omega_{\mathcal{L}}}$. This implies that $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$ for all $t \in]\bar{t} - \eta, \bar{t}]$, thus contradicting $\bar{t} = t_{\mathcal{L}}$ as the infimum in (2.13).

We next show that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap \partial C_{\omega_{\mathcal{L}}} \neq \emptyset$. Suppose that instead $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap \partial C_{\omega_{\mathcal{L}}} = \emptyset$. For notation, let $\hat{y}_0 := \hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z})$. Since A is compact, $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z})$ is compact for all t and $\mathbf{z}$. Now consider the distance $\gamma := d(\partial C_{\omega_{\mathcal{L}}}, \hat{y}_0)$. If $\gamma = 0$, then the intersection must be non-empty, as both sets are closed. Therefore, we assume that $\gamma > 0$. By the upper hemi-continuity of the set-valued flow, there exists $\eta > 0$ such that for all $t \in [\bar{t}, \bar{t} + \eta[$, we have that $\phi(t, \hat{y}_0, S(U)) \subset B(\hat{y}_0, \gamma/2)$. By the definition of the closed-loop information state, for all $t \geq \bar{t}$ we have that $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \subset \phi(t, \hat{y}_0, S(U))$. This implies that for all $t \in [\bar{t}, \bar{t} + \eta[$ we have $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} = \emptyset$, since $d(\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}), C_{\omega_{\mathcal{H}}}) > \gamma/2 > 0$. This contradicts $\bar{t} = t_{\mathcal{L}}$ as given in equation (2.13), hence we must have that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap \partial C_{\omega_{\mathcal{L}}} \neq \emptyset$.

We have thus shown that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$, $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap \partial C_{\omega_{\mathcal{L}}} \neq \emptyset$ and $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$. From the definition of the modified dynamic set-valued feedback map $\hat{G}$ given in (2.12), we must necessarily have that $\bar{\omega}^{cl}(\bar{t}) = \omega_{\mathcal{L}} = \hat{G}(\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}))$. From definitions (2.13) and (2.14), we therefore have that $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} \neq \emptyset$ and $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} \neq \emptyset$ for all $t \in [\bar{t}, t_1]$. Therefore, by the definition of $\hat{G}$ in equation (2.12), we have that $\bar{\omega}^{cl}(t) = \omega_{\mathcal{L}} = \hat{G}(\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}))$ for all $t \in [\bar{t}, t_1]$. Let $v \in \hat{y}^{cl}(t_1, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}}$ and choose $w \in \hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z})$ such that $\phi(t_1 - \bar{t}, w, \varphi(\omega_{\mathcal{L}}, \mathbf{d})) = v$ for some $\mathbf{d} \in S(\Delta)$ (note that such a w exists by the definition of the information state $\hat{y}$). Since $v \in C_{\omega_{\mathcal{L}}}$ and $\omega(t) = \omega_{\mathcal{L}}$ for all $t \in [\bar{t}, t_1]$, we must have that $w \in C_{\omega_{\mathcal{L}}}$ by the definition of $C_{\omega_{\mathcal{L}}}$. This leads to a contradiction, since we assumed that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$. As a consequence, such a time t_1 for which Case(I) holds cannot exist.

For Case(II), an equivalent argument holds by interchanging $\omega_{\mathcal{L}}$ with $\omega_{\mathcal{H}}$, and $C_{\omega_{\mathcal{L}}}$ with $C_{\omega_{\mathcal{H}}}$, then showing that this leads to a contradiction of $t_{\mathcal{H}}$ as defined in (2.14).

For Case(III), the argument is similar. First, it can be shown that $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap \partial C_{\omega_{\mathcal{L}}} \neq \emptyset$ and $\hat{y}^{cl}(\bar{t}, A, \bar{\omega}^{cl}, \mathbf{z}) \cap \partial C_{\omega_{\mathcal{H}}} \neq \emptyset$ by a continuity argument (similar to the one made in Case(I)). The proof proceeds as in Case(I) with the eventual contradiction regarding the definition $C_{\omega_{\mathcal{L}}}$, and thus contradicting the existence of $t_{\mathcal{L}}$ and $t_{\mathcal{H}}$ as defined in (2.13) and (2.14) respectively.

Therefore $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$ must hold for all $t \in \mathbb{R}_+$ under any control $\bar{\omega}^{cl} \in S(\Omega)$ generated by $(\mathscr{G}, \hat{G})$. From the definition of G in (2.11), it must be that $G(\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z})) = \hat{G}(\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}))$ for all $t \in \mathbb{R}_+$. This implies that for every closed-loop information state $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z})$ and feedback ω^{cl} generated by the control system $(\mathscr{G}, G)$, there is a corresponding feedback $\bar{\omega}^{cl}$ and closed-loop information state $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z})$ generated by the control system $(\mathscr{G}, \hat{G})$ such that $\bar{\omega}^{cl} = \omega^{cl}$ and $\hat{y}^{cl}(t, A, \bar{\omega}^{cl}, \mathbf{z}) = \hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z})$. This implies that $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap C_{\omega_{\mathcal{L}}} = \emptyset$ for all $t \in \mathbb{R}_+$. Therefore, the closed-loop information state generated by the control system $(\mathscr{G}, G)$ satisfies $\hat{x}^{cl}(t, A, \omega^{cl}, \mathbf{z}) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$. □

We can thus summarize the solutions to Problem1 and Problem 2 in the two following theorems, respectively.

Theorem 3. *(Solution to Problem 1) For an order preserving game structure* $\mathscr{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$*, a convex set* $\hat{x}_0 \subset X$ *is in* $\bar{\mathcal{W}}$ *if and only if* $\hat{x}_0 \cap C_{\omega_{\mathcal{H}}} = \emptyset$ *or* $\hat{x}_0 \cap C_{\omega_{\mathcal{L}}} = \emptyset$*. Furthermore, if* $\hat{x}_0 \in \bar{\mathcal{W}}$ *is also compact, then a dynamic feedback map* $G : 2^X \rightrightarrows \Omega$ *is given by (2.11).*

Proof. By Theorem 1, there exists a control signal $\omega \in S(\Omega)$ such that $\phi(t, \hat{x}_0, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \cap \mathbf{B} = \emptyset$ for all $\mathbf{d} \in S(\Delta)$ and all $t \in \mathbb{R}_+$ if and only if $\hat{x}_0 \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $\hat{x}_0 \cap C_{\omega_{\mathcal{L}}} = \emptyset$.

Assuming that $\mathbf{z}$ is the worst-case observation signal, that is, $\mathbf{z}(t) = \bar{z}$ for all $t \in \mathbb{R}_+$, we have that $\hat{x}(t, \hat{x}_0, \omega, \mathbf{z}) = \bigcup_{\mathbf{d} \in S(\Delta)} \phi(t, \hat{x}_0, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d}))$ for all $t \in \mathbb{R}_+$. Therefore, there is a control signal $\omega \in S(\Omega)$ such that $\hat{x}(t, \hat{x}_0, \omega, \mathbf{z}) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$ if and only if $\hat{x}_0 \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $\hat{x}_0 \cap C_{\omega_{\mathcal{L}}} = \emptyset$. By the definition of $\bar{\mathcal{W}}$, we thus have that $\hat{x}_0 \in \bar{\mathcal{W}}$ if and only if $\hat{x}_0 \cap C_{\omega_{\mathcal{H}}} = \emptyset$ or $\hat{x}_0 \cap C_{\omega_{\mathcal{L}}} = \emptyset$. Since the set of initial conditions $\hat{x}_0$ is compact, Theorem 2 further shows that the feedback map G given by expression (2.11) maintains $\hat{x}(t, \hat{x}_0, \omega, \mathbf{z})$ with $\omega(\tau) \in G(\hat{x}(\tau, \hat{x}_0, \omega, \mathbf{z}))$ for all $\tau \in \mathbb{R}_+$ not intersecting $\mathbf{B}$ for all $t \in \mathbb{R}_+$. □

Theorem 4. *(Solution to Problem 2) For an order preserving game structure* $\mathcal{G} = (\Sigma, \Omega, \Delta, \varphi, \mathbf{B})$*, the set* $\mathcal{W}$ *of Problem 2 is given by* $\mathcal{W} = X \backslash (C_{\omega_{\mathcal{H}}} \cap C_{\omega_{\mathcal{L}}})$*. A feedback map* $g : X \rightrightarrows \Omega$ *is given by*

$$g(x) := \begin{cases} \omega_{\mathcal{H}} & \text{if } x \in C_{\omega_{\mathcal{L}}} \text{ and } x \in \partial C_{\omega_{\mathcal{H}}} \\ \omega_{\mathcal{L}} & \text{if } x \in C_{\omega_{\mathcal{H}}} \text{ and } x \in \partial C_{\omega_{\mathcal{L}}} \\ \omega_{\mathcal{L}} & \text{if } x \in \partial C_{\omega_{\mathcal{H}}} \text{ and } x \in \partial C_{\omega_{\mathcal{L}}} \\ \Omega & \text{otherwise.} \end{cases}$$

Proof. Direct consequence of Corollary 1 and Theorem 2, in which A is a singleton. □

The order preserving properties of the dynamics allow for the construction of discrete-time linear complexity algorithms for the computation of the restricted capture sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$. These algorithms are presented in the next section.

2.6 Algorithms

By virtue of Theorems 3 and 4, the dynamic and static control Problems 1 and 2 can be solved by only computing the sets $C_{\omega_{\mathcal{H}}}$ and $C_{\omega_{\mathcal{L}}}$. For a class of order preserving systems in discrete-time, we introduce an algorithm for computing the restricted capture set C_ω. This algorithm has linear complexity with respect to the number of continuous variables.

The restrictions on the game structure $\mathcal{G}$ imposed are:
Assumption (a) $f^i(x^i, u^i)$ has no dependency on x_1^i;

Assumption (b) The bad set $\mathbf{B}$ is given by $\mathbf{B} := \{x \in X \mid (x_1^1, x_1^2) \in B\}$,

with $B :=]L, H[\subset \mathbb{R}^2$.

This structure of $f^i(x^i, u^i)$ is found, for example, in vector fields derived from Newton's laws with no position dependent forces (such as gravity). The bad set $\mathbf{B}$ generated by the open rectangle set B can represent, for example, the set of all collision configurations between two agents evolving on intersecting paths. If B is a more general bounded o.p.c. set, a rectangular over-approximation can be employed.

2.6.1 Discrete-Time Model

Seeking digital implementation, we illustrate the algorithm in discrete-time. For agent $i \in \{1, 2\}$, denote the state space $\bar{X}^i := X_2^i \times \ldots \times X_n^i$, the corresponding state $\bar{x}^i \in \bar{X}^i$, and the set of discrete-time signals $D : \mathbb{N} \to U^i$ as $D(U^i)$. Define the discretization of the system (employing forward Euler approximation) for agent $i \in \{1, 2\}$ with step size $\Delta T > 0$, input $\mathbf{u}^i \in D(U^i)$ and step $n \in \mathbb{N}$ as

$$x^i[n+1] = x^i[n] + \Delta T f^i(x^i[n], \mathbf{u}^i[n]).$$

For the index $n \in \mathbb{N}$, initial condition $x^i \in X^i$, and input signal $\mathbf{u}^i \in D(U^i)$, we denote the discrete-time flow $\Phi^i : \mathbb{N} \times X^i \times D(U^i) \to X^i$ as $\Phi^i(n, x^i, \mathbf{u}^i)$, which satisfies

$$\Phi^i(n+1, x^i, \mathbf{u}^i) = \Phi^i(n, x^i, \mathbf{u}^i) + \Delta T f^i(\Phi^i(n, x^i, \mathbf{u}^i), \mathbf{u}^i[n-1]) \text{ for all } n \in \mathbb{N}, \tag{2.15}$$

where $\Phi^i(0, x^i, \mathbf{u}^i) = x^i$. We assume the discrete flow Φ^i is continuous with respect to input $\mathbf{u}^i \in D(U^i)$. Let $\mathbf{z}^i \in D(O)$ be the output measurement. From Definition 4, the output map is given by $h^i(z^i[n]) = [\inf h^i(z^i[n]), \sup h^i(z^i[n])]$. The j^{th} component of the flow is denoted as $\Phi_j^i(n, x^i, \mathbf{u}^i)$

For the parallel composition of two systems $\Sigma = \Sigma^1 \| \Sigma^2$, the discretization and discrete-time flow extend to

$$\begin{aligned} \Delta T f(x[n], \mathbf{u}[n]) &:= (\Delta T f^1(x^1[n], \mathbf{u}^1[n]), \Delta T f^2(x^2[n], \mathbf{u}^2[n])) \\ \Phi(n, x, \mathbf{u}) &:= (\Phi^1(n, x^1, \mathbf{u}^1), \Phi^2(n, x^2, \mathbf{u}^2)). \end{aligned}$$

The game input map, as in Definition 7, easily extends to discrete-time control signals $\omega \in D(\Omega)$ and disturbance signals $\mathbf{d} \in D(\Delta)$ as $\mathbf{u}[n] = \varphi(\omega[n], \mathbf{d}[n])$.

From Assumption (a), it follows that for an initial condition $(x_1, \bar{x}) \in X$ and input $\mathbf{u} \in D(U)$, we have that

$$\Phi_1(n, x, \mathbf{u}) = x_1 + \Phi_1(n, (0, \bar{x}), \mathbf{u}) \text{ for all } n \in \mathbb{N}, \tag{2.16}$$

where the state $(0, \bar{x})$ represents the initial condition x with the state x_1 set to zero. This property implies that the flow projected onto the subspace X_1 has no dependency on the state x_1 other than the initial condition.

2.6.2 Restricted Capture Set C_ω Computation

The definition of the discrete-time capture set is the same as in continuous time, however now the index $n \in \mathbb{N}$ replaces time $t \in \mathbb{R}_+$, and the discrete signal $\mathbf{d} \in D(\Delta)$ replaces the continuous signal $\mathbf{d} \in S(\Delta)$. This is mathematically represented as

$$C_\omega = \{x \in X \mid \exists\, n \in \mathbb{N},\ \exists\, \mathbf{d} \in D(\Delta) \text{ s.t. } \Phi(n, x, \varphi(\omega, \mathbf{d})) \in \mathbf{B}\}.$$

To compute the restricted capture set, we introduce the sequences $\{L^i(n, x^i, \omega^i)\}, \{H^i(n, x^i, \omega^i)\} \subset X_1^i$ generated with the state $x^i \in X^i$ and constant control input $\omega^i \in D(\Omega^i)$. These sequences are defined as

$$\begin{aligned} L^i(n, x^i, \omega^i) &:= L^i - \Phi_1^i(n, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_H^i)) \\ H^i(n, x^i, \omega^i) &:= H^i - \Phi_1^i(n, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_L^i)). \end{aligned}$$

We can combine these sequences for $i \in \{1, 2\}$ and define $L(n, x, \omega) := (L^1(n, x^1, \omega^1), L^2(n, x^2, \omega^2))$, $H(n, x, \omega) := (H^1(n, x^1, \omega^1), H^2(n, x^2, \omega^2))$.

The sequence $\{L(n, x, \omega)\}_{n\in\mathbb{N}}$ represents the backward integration of L with state $(0, \bar{x})$, control input ω and constant disturbance input $\mathbf{d}_H$. The sequence $\{H(n, x, \omega)\}_{n\in\mathbb{N}}$ represents the backward integration of H with state $(0, \bar{x})$, control input ω and constant disturbance input $\mathbf{d}_L$. We use both these sequences to define a sequence of rectangle sets as $\{]L(n, x, \omega), H(n, x, \omega)[\}_{k\in\mathbb{N}} \subset \mathbb{R}^2$.

We introduce Algorithm 1, which can be used to compute the restricted capture set C_ω, by recursively computing the elements of the sequence $\{]L(n, x, \omega), H(n, x, \omega)[\}_{n\in\mathbb{N}}$. To accommodate the case of state uncertainty (§2.6.3), the input of Algorithm 1 is a *set* $\hat{x} \subset X$ rather than a singleton $x \in X$.

Algorithm 1 $\tilde{C}_\omega = \text{CaptureSetSlice}(\hat{x}, \omega)$

Input: $(\hat{x}, \omega) \in 2^X \times D(\Omega)$

$n = 1$
loop
  Termination met when the sequence $H(n, \inf \hat{x}, \omega)$ *is no longer in the set* $Cone_+(\inf \hat{x}_1)$.
  if $\inf \hat{x}_1 \leq H(n, \inf \hat{x}, \omega)$ and $\inf \hat{x}_1 \notin]L(n, \sup \hat{x}, \omega), H(n, \inf \hat{x}, \omega)[$ **then**
    $n = n + 1$
  else
    return $\tilde{C}_\omega = \bigcup_{k\leq n}]L(k, \sup \hat{x}, \omega), H(k, \inf \hat{x}, \omega)[$.
  end if
end loop

Output: $\tilde{C}_\omega \subset X_1$.

We can interpret Algorithm 1 as the backward propagation of the rectangle set $]L, H[$ with control signal ω and all disturbances. This, in turn, by the order preserving properties of the discrete-time flow with respect to the input, only requires the upper bound $\mathbf{d}_H$ and the lower bound $\mathbf{d}_L$. To show *termination* of Algorithm 1, we note that condition (iii) of Definition 4 implies that the sequence $\{H(n, x, \omega)\}_{n\in\mathbb{N}}$ is strictly monotonically decreasing without limit for any $x \in X$ and $\omega \in D(\Omega)$. Therefore, there must be some finite $n \in \mathbb{N}$

such that $\inf \hat{x}_1 \not\leq H(n, \inf \hat{x}, \omega)$, implying termination of Algorithm 2.

Claim 1.

$$C_\omega = \left\{ x \in X \mid x_1 \in \tilde{C}_\omega = \text{CaptureSetSlice}(\{x\}, \omega) \right\}.$$

Proof. Denote $S := \left\{ x \in X \mid x_1 \in \tilde{C}_\omega = \text{CaptureSetSlice}(\{x\}, \omega) \right\}$. We show first that $C_\omega \subseteq S$ and then that $C_\omega \supseteq S$.

($\subseteq$) Let $x \in C_\omega$, then by the definition of C_ω we have that there is $\mathbf{d} \in D(\Delta)$ and $\bar{n} \in \mathbb{N}$ such that $L \leq \Phi_1(\bar{n}, x, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \leq H$. From equation (2.16), we have that

$$L - \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \leq x_1 \leq H - \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})). \tag{2.17}$$

From the order preserving property of the game input map with respect to the disturbance and by the order preserving property of the discrete-time flow with respect to the input, we have that

$$\Phi_1(\bar{n}, (0, \bar{x}), \varphi(\omega, \mathbf{d}_{\mathcal{L}})) \leq \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \leq \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\omega, \mathbf{d}_{\mathcal{H}})). \tag{2.18}$$

Therefore, from expressions (2.17) and (2.18), we have that

$$\begin{aligned} x_1 \leq H - \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) &\leq H - \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\omega, \mathbf{d}_{\mathcal{L}})) = H(\bar{n}, x, \omega) \\ x_1 \geq L - \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) &\geq L - \Phi_1(\bar{n}, (0, \bar{x}), \varphi(\omega, \mathbf{d}_{\mathcal{H}})) = L(\bar{n}, x, \omega), \end{aligned}$$

which imply $x \in S$.

($\supseteq$) Let $x \in S$, for agent $i \in \{1, 2\}$ we have that $x_1^i \leq H^i(\bar{n}, x^i, \omega^i) = H^i - \Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_L^i))$ and $x_1^i \geq L^i(\bar{n}, x^i, \omega^i) = L^i - \Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_H^i))$ for some $\bar{n} \in \mathbb{N}$. We can rearrange these inequalities to give $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_L^i)) \leq H^i - x_1^i$ and $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_H^i)) \geq L^i - x_1^i$. If either $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_L^i)) \geq L^i - x_1^i$ or $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_H^i)) \leq H^i - x_1^i$, we have that there is a disturbance $\mathbf{d}$ such that $x_1^i + \Phi_1^i(\bar{n}, (0, \bar{x}), \varphi^i(\omega^i, \mathbf{d}^i))) = \Phi_1^i(\bar{n}, x^i, \varphi(\boldsymbol{\lambda}, \mathbf{u}, \mathbf{d})) \in]L^i, H^i[$. If neither of these two cases

is satisfied, the following inequalities are satisfied: $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi(\omega^i, \mathbf{d}_L^i)) < L - x_1$ and $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}_H^i)) > H - x_1$. Since $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \cdot)) : D(\Delta^i) \to X_1^i$ is a continuous function and $D(\Delta^i)$ is a connected metric space with $\Delta^i = [\delta_L^i, \delta_H^i]$, by the intermediate value theorem there must be $\mathbf{d}^i \in D(\Delta^i)$ such that $\Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}^i)) = w \in]L^i - x_1^i, H^i - x_1^i[$. As a consequence, for such a $\mathbf{d}^i$ we have that $x_1^i + \Phi_1^i(\bar{n}, (0, \bar{x}^i), \varphi^i(\omega^i, \mathbf{d}^i)) = \Phi_1^i(\bar{n}, x^i, \varphi^i(\omega^i, \mathbf{d}^i)) \in]L^i, H^i[$. Since this holds for arbitrary $i \in \{1, 2\}$, we have shown that $x \in C_\omega$. □

Note that the sets C_ω are $2n$ dimensional. Claim 1 shows that these high dimensional sets can be computed by just computing a sequence of lower $\{L(n, x, \omega)\}_{n\in\mathbb{N}}$ and upper $\{H(n, x, \omega)\}_{n\in\mathbb{N}}$ bounds in X_1, which are parameterized by the $2n$ state variables x. For any fixed value of $x \in X$, the union of intervals $\cup_{n\in\mathbb{N}}]L(n, x, \omega), H(n, x, \omega)[$ over all $n \in \mathbb{N}$ represents the two dimensional slice of C_ω corresponding to the state x.

The boundary of the capture set ∂C_ω must be reinterpreted, as now the discrete-time flow can enter the interior of the capture set without touching the boundary. We provide a definition of the capture set boundary ∂C_ω as

$$\partial C_\omega := \{x \in X \backslash C_\omega \mid \exists\, \delta \in \Delta \text{ s.t. } x + \Delta T f(x, \varphi(\omega, \delta)) \in C_\omega\}. \tag{2.19}$$

According to this definition, a state outside of the restricted capture set is said to be on the boundary of the restricted capture set, if there is some disturbance such that the state is mapped inside the capture set in one step.

2.6.3 Dynamic Feedback Implementation

Since the dynamics of the system are order preserving with respect to the state and to the input, we construct a state estimator that keeps track of only the lower and upper bounds of the information state similar to the estimator proposed in [37]. Let $\vee\hat{x} := \sup \hat{x}$ and $\wedge\hat{x} := \inf \hat{x}$ denote the upper and lower bounds, respectively, of the set of possible current

states $\hat{x}$ (the sup and inf are taken component-wise in accordance to the partial ordering defined on $(X, \leq)$). Then, a state estimate $\hat{x}[n]$ is constructed with Algorithm 2, by only updating the upper and lower bounds of $\hat{x}[n-1]$. To construct the state estimate, first the previous state estimate is mapped forward under the discrete update map with the control input supplied and all possible disturbances. Then, the measurement is used to further restrict the set of all possible compatible states. Conditions leading to estimator convergence are provided in [37] for a class of systems.

Algorithm 2 $\hat{x}[n] = \text{StateEstimate}(\hat{x}[n-1], \omega[n-1], z[n])$

Input: $(\hat{x}[n-1], z[n]) \in 2^X \times O$

Update state estimate.
$\vee\hat{x}[n] = \inf\{\Delta T f(\vee\hat{x}[n-1], \varphi(\omega[n-1], \delta_H)), \sup h(z[n])\}$.
$\wedge\hat{x}[n] = \sup\{\Delta T f(\wedge\hat{x}[n-1], \varphi(\omega[n-1], \delta_L)), \inf h(z[n])\}$.

Return state estimate with upper and lower bounds.
return $\hat{x}[n] = [\wedge\hat{x}[n], \vee\hat{x}[n]]$.

Output: $\hat{x}[n] \subset X$.

To implement the closed-loop feedback $G : 2^X \rightrightarrows U$ given by equation (2.11) from §2.5.2, one must check whether the state estimate $\hat{x}[n]$ intersects C_{ω_H} and C_{ω_L}. Since the sequence $L(k, x, \omega)$ is order reversing in the argument x, a sufficient condition guaranteeing that $\hat{x}[n] \cap C_\omega = \emptyset$ is that

$$\hat{x}_1[n] \cap \bigcup_{k\in\mathbb{N}}]L(k, \vee\hat{x}[n], \omega), H(k, \wedge\hat{x}[n], \omega)[= \emptyset. \tag{2.20}$$

We introduce Algorithm 3, which can be used to compute the feedback $\omega[n]$ generated by the set-valued map G by using the current state $\hat{x}[n]$ and the state prediction $\hat{x}[n+1]$.

We can interpret Algorithm 3 as the discrete-time implementation of the set-valued map G, as defined in (2.11). The algorithm is comprised of a series of steps. First, capture set slices are constructed with Algorithm 1 for the state prediction. If the state prediction $\hat{x}[n+1]$ has non-empty intersection with each restricted capture set, as established by

Algorithm 3 $\omega = \text{FeedbackMap}(\hat{x}[n+1], \hat{x}[n])$

Input: $(\hat{x}[n+1], \hat{x}[n]) \in 2^X \times 2^X$

Construct capture set slices for state prediction.
$\tilde{C}_{\omega_{\mathcal{L}}} = \text{CaptureSetSlice}(\hat{x}[n+1], \omega_{\mathcal{L}})$, $\tilde{C}_{\omega_{\mathcal{H}}} = \text{CaptureSetSlice}(\hat{x}[n+1], \omega_{\mathcal{H}})$

Check if predicted state $\hat{x}[n+1]$ intersects both capture set slices.
if $\hat{x}[n+1] \cap \tilde{C}_{\omega_{\mathcal{L}}} \neq \emptyset$ *and* $\hat{x}[n+1] \cap \tilde{C}_{\omega_{\mathcal{H}}} \neq \emptyset$ **then**

Construct capture set slices for current state.
$\tilde{C}_{\omega_{\mathcal{L}}} = \text{CaptureSetSlice}(\hat{x}[n], \omega_{\mathcal{L}})$, $\tilde{C}_{\omega_{\mathcal{H}}} = \text{CaptureSetSlice}(\hat{x}[j], \omega_{\mathcal{H}})$

Determine control according to equation (2.11).
if $\hat{x}_1[n] \cap \tilde{C}_{\omega_{\mathcal{L}}} = \emptyset$ *and* $\hat{x}_1[n] \cap \tilde{C}_{\omega_{\mathcal{H}}} \neq \emptyset$ **then**
$\omega = \omega_{\mathcal{L}}$
else if $\hat{x}_1[n] \cap \tilde{C}_{\omega_{\mathcal{L}}} \neq \emptyset$ *and* $\hat{x}_1[n] \cap \tilde{C}_{\omega_{\mathcal{H}}} = \emptyset$ **then**
$\omega = \omega_{\mathcal{H}}$
else
$\omega = \omega_{\mathcal{L}}$
end if

else
No control specified.
$\omega \in \Omega$
end if

Output: $\omega \subset \Omega$.

equation (2.20), then the state estimate $\hat{x}[n]$ either has non-empty intersection or is on the boundary of each restricted capture set. The state estimate $\hat{x}[n]$ is on the boundary of a restricted capture set, as defined in (2.19), if the state estimate $\hat{x}[n]$ has empty intersection with the corresponding capture set slice constructed with Algorithm 1. If the intersection is non-empty, then the state estimate $\hat{x}[n]$ has non-empty intersection with the restricted capture set. Lastly, control is evaluated with the set-valued map G based on the restricted capture set membership established.

The closed-loop control system is implemented with Algorithm 4, where the feedback and state estimate are given by $(\omega[n], \hat{x}[n]) = \text{ControlSystem}(\hat{x}[n-1], \mathbf{z}[n])$. We can summarize Algorithm 4 as follows. First, the state estimate is constructed with Algorithm 2. Next, a state prediction is constructed by mapping the current state estimate forward

with all possible disturbance signals. Finally, control is evaluated with Algorithm 3 based on current state estate estimate and state prediction.

Algorithm 4 $(\omega^{cl}[n], \hat{x}[n]) = \text{ControlSystem}(\hat{x}[n-1], \mathbf{z}[n])$

Input: $(\hat{x}[n-1], \mathbf{z}[n]) \in 2^X \times O$

Update state estimate.
$\hat{x}[n] = \text{StateEstimate}(\hat{x}[n-1], \mathbf{z}[n])$

Construct state prediction.
$\hat{x}[n+1] = [\Delta T\, f(\vee \hat{x}[n], \varphi(\omega[n], \delta_L)), \Delta T\, f(\wedge \hat{x}[n], \varphi(\omega[n], \delta_H))]$

Compute closed-loop feedback.
$\omega^{cl}[n] = \text{FeedbackMap}(\hat{x}[n+1], \hat{x}[n])$

Output: $(\omega^{cl}[n], \hat{x}[n]) \in \Omega \times 2^X$

2.7 Simulation and Experimental Results

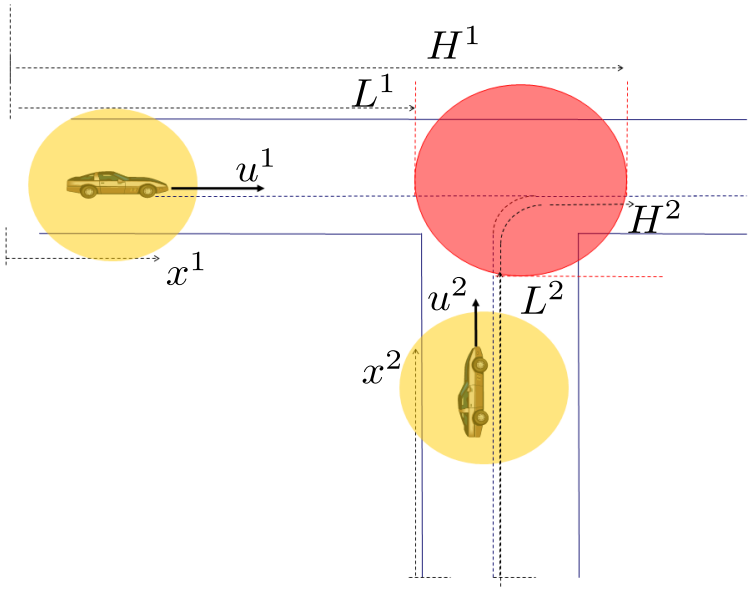


Figure 2.7: Vehicles approaching a "T" intersection. A collision occurs if two vehicles are in the set B at the same time.

In this section, we illustrate the application of the algorithms outlined in §2.6 to the two-vehicle collision avoidance problem introduced in §2.1, in which we now consider disturbances, imperfect state information, and higher order piecewise continuous vehicle dynamics.

In-vehicle cooperative active safety and related technologies continue to be examined world-wide by government and industry consortium, such as the Crash Avoidance Metrics Partnership (CAMP) [2], the Vehicle Infrastructure Integration Consortium (VIIC) [5, 6] in the U.S., the Car2Car Communications Consortium in Europe [1], the Advanced Safety Vehicle project 3 (ASV3) in Japan, and by university research centers such as the Virginia Tech Transportation Institute (VTTI) and the California PATH. In the near future, ITS is expected to become more comprehensive by connecting vehicles with each other and with the surrounding road infrastructure through vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) wireless communication.

Here, we consider three different scenarios. In the first scenario, the cooperative case, we assume V2V communication. The two vehicles thus share information and cooperate to prevent a potential collision. In the second scenario, the competitive case, we assume that the two vehicles cannot communicate with each other, for example, only one of the two vehicles is equipped with the on-board active safety system. This scenario is of high interest, as any realistic deployment of cooperative active safety systems will not be universally equipped on all vehicles. The third scenario assumes V2V communication and thus cooperation between the two vehicles. However, we assume that the dynamic model of the vehicles is subject to modeling uncertainty. For this combined case, experimental results on a concrete in-lab implementation are presented. In all of these three cases, we consider the traffic intersection instance depicted in Figure 2.7 as a reference.

The longitudinal dynamics of each vehicle along its path can be modeled employing Newton's laws. Let $p \in \mathbb{R}$ denote the longitudinal displacement along the vehicle path. The longitudinal vehicle dynamics can thus be written as

$$\ddot{p} = [\mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)](f_w - f_{brake} - \frac{\rho_{air}}{2} C_D A_f v^2 - C_{rr}\mathcal{M}g - \mathcal{M}g\sin(\theta_{\text{road}})),$$

in which $\mathcal{R}$ is the tire radius, J_w is the wheel inertia, $\mathcal{M}$ is the mass of the vehicle, $f_w = \tau_w \mathcal{R}$

where τ_w is the drive shaft output torque, f_{brake} is the brake force, ρ_{air} is the air density, C_D is the drag coefficient, A_f is the projected front area of the vehicle, v is the longitudinal vehicle velocity, C_{rr} is the rolling resistance coefficient, g is the gravity constant, and θ_{road} is the road gradient. For more details on this model, the reader is referred to [97] and to the references therein. For automatic driving, f_w and f_{brake} are control inputs to the longitudinal dynamics of the vehicle. Assuming that the road is flat and that the air drag term is negligible, we can re-write the longitudinal dynamics as

$$\ddot{p} = a\,u + b, \tag{2.21}$$

in which $u = f_w - f_{brake}$ is the total force, which is the control input to the vehicle, $a = \mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)$, and $b = -\mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)\, C_{rr}\mathcal{M}g$.

For vehicle $i \in \{1, 2\}$, we denote (see Figure 2.7) the longitudinal displacement along its path by x_1^i and the longitudinal speed by x_2^i. As a consequence, the longitudinal dynamics for vehicle $i \in \{1, 2\}$ can be re-written as

$$\begin{aligned} \dot{x}_1^i &= x_2^i \\ \dot{x}_2^i &= a^i u^i + b^i. \end{aligned}$$

In order to prevent the vehicle from stopping (to prevent the trivial solution in which the vehicles come to a stop) and from exceeding a maximum speed (to respect road speed limitations), we consider the hybrid system depicted in Figure 2.8. For each vehicle sub-

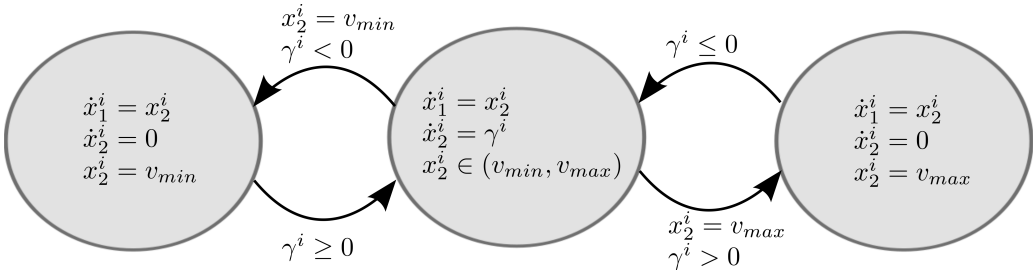


Figure 2.8: Hybrid system modeling the vehicle system Σ^i for $i \in \{1, 2\}$. In the diagram, we have defined $\gamma^i := a^i u^i + b^i$.

system Σ^i, we choose for $z^i \in \mathbb{R}^2$ an output map $h(z^i) = [z_1^i - d_1, z_1^i + d_1] \times [z_2^i - d_2, z_2^i + d_2]$ (a continuous set-valued function), in which z^i is a pair of position/speed measurements assumed to be continuous in time, d_1 models uncertainty on the position measurement, and d_2 models uncertainty on the speed measurement. While d_2 is practically close to zero as the on-board speed measurements are quite accurate, d_1 can be quite large due to GPS positioning error. One can verify that systems Σ^i are order preserving systems, and the differential inclusion generated by all inputs is Marchaud.

The corresponding discrete-time dynamical system with time step ΔT is given by

$$\begin{aligned} x_1^i[n+1] &= x_1^i[n] + \Delta T x_2^i[n] \\ x_2^i[n+1] &= x_2^i[n] + \Delta T \gamma^i, \end{aligned}$$

in which $\gamma^i = a^i u^i + b^i$ in the central mode of Figure 2.8 and $\gamma^i = 0$ in the right and left modes of the same figure.

The bad set **B** is constructed with the rectangle set $B =]L^1, H^1[\times]L^2, H^2[$.

2.7.1 The Cooperative Case

In the cooperative case, we have that $u = (u^1, u^2) = \varphi_{\text{coop}}(\omega, \delta) = (\omega^1, \omega^2)$, that is, both of the agents are controlled and $(u^1, u^2) \in \Omega = [\omega_L^1, \omega_H^1] \times [\omega_L^2, \omega_H^2]$. We implement the algorithms of §2.6 to compute the restricted capture sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$. Figure 2.9 shows snapshots in the position plane of the trajectory of the set $[\wedge\hat{x}, \vee\hat{x}]$ for the closed-loop system. As soon as the set $[\wedge\hat{x}, \vee\hat{x}]$ hits the intersection of the two restricted capture sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$, the safety control acts and, as a result, set $[\wedge\hat{x}, \vee\hat{x}]$ slides along the boundary of the capture set until it passes **B**. Note that the sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$ are each four dimensional. The plots of Figure 2.9 show slices of such sets in the position plane corresponding to the value of the current speeds.

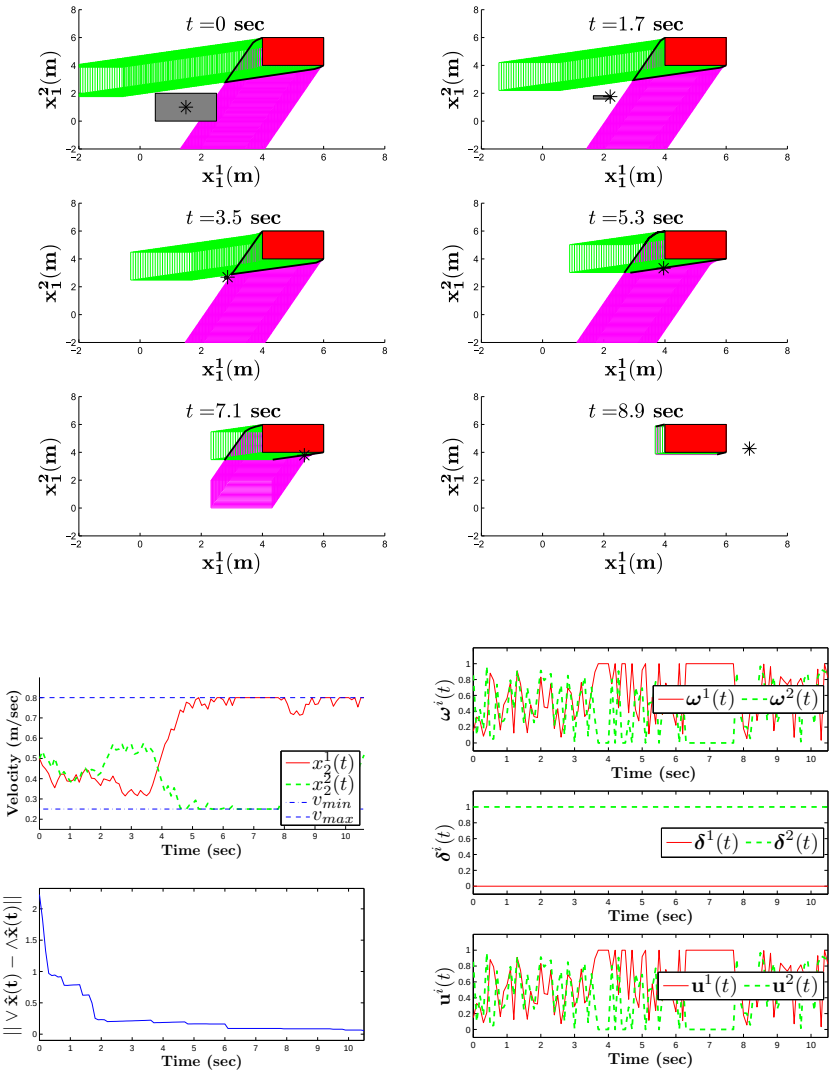


Figure 2.9: The cooperative case. The above plots depict snapshots of the dynamic evolution of the closed-loop system. The system considered has $a^i = 1$ and $b^i = -.5$ for $i \in \{1, 2\}$, with $v_{min} = .25$m/sec and $v_{max} = .8$m/sec. We choose $\Delta T = .1$ sec, $\mathbf{B} =]4, 6[\times\mathbb{R}\times]4, 6[\times\mathbb{R}$, $\Omega = [0, 1]\times[0, 1]$, $x_0 = (1.5, .5, 1, .5)$, $\hat{x}_0 = [.5, 2.5]\times[.4, .6]\times[0, 2]\times[.4, .6]$. The measurements z are generated randomly with a uniform probability distribution in the interval $[x(t) - (1, .1, 1, .1), x(t) + (1, .1, 1, .1)]$ so that $h(z) = [z - (1, .1, 1, .1), z + (1, .1, 1, .1)]$. The grey box represents the projection of $\hat{x}(t)$ onto the (x_1^1, x_1^2) plane, with the black asterisk representing the state of the system projected onto the (x_1^1, x_1^2) plane. The red box represents the projection of $\mathbf{B}$ onto the (x_1^1, x_1^2) plane, the slice of $C_{\omega_{\mathcal{L}}}$ corresponding to the current speeds is shown in green and the slice of $C_{\omega_{\mathcal{H}}}$ corresponding to the current speeds is shown in purple. Plots of the velocities, controls, disturbances, estimation error $\| \wedge \hat{x} - \vee\hat{x}\|$, and inputs are depicted in the lower panels.

2.7.2 The Competitive Case

In the competitive case, we have that $u = (u^1, u^2) = \varphi(\omega, \delta) = (\omega^1, \delta^2)$, that is, the first agent is controlled while the second one is not and $(u^1, u^2) \in [\omega_L^1, \omega_H^1] \times [\delta_L^2, \delta_H^2]$. We

implement the algorithms of §2.6 to compute the restricted capture sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$. Figure 2.10 shows snapshots in the position plane of the trajectory of the set $[\wedge\hat{x}, \vee\hat{x}]$ for the closed-loop system.

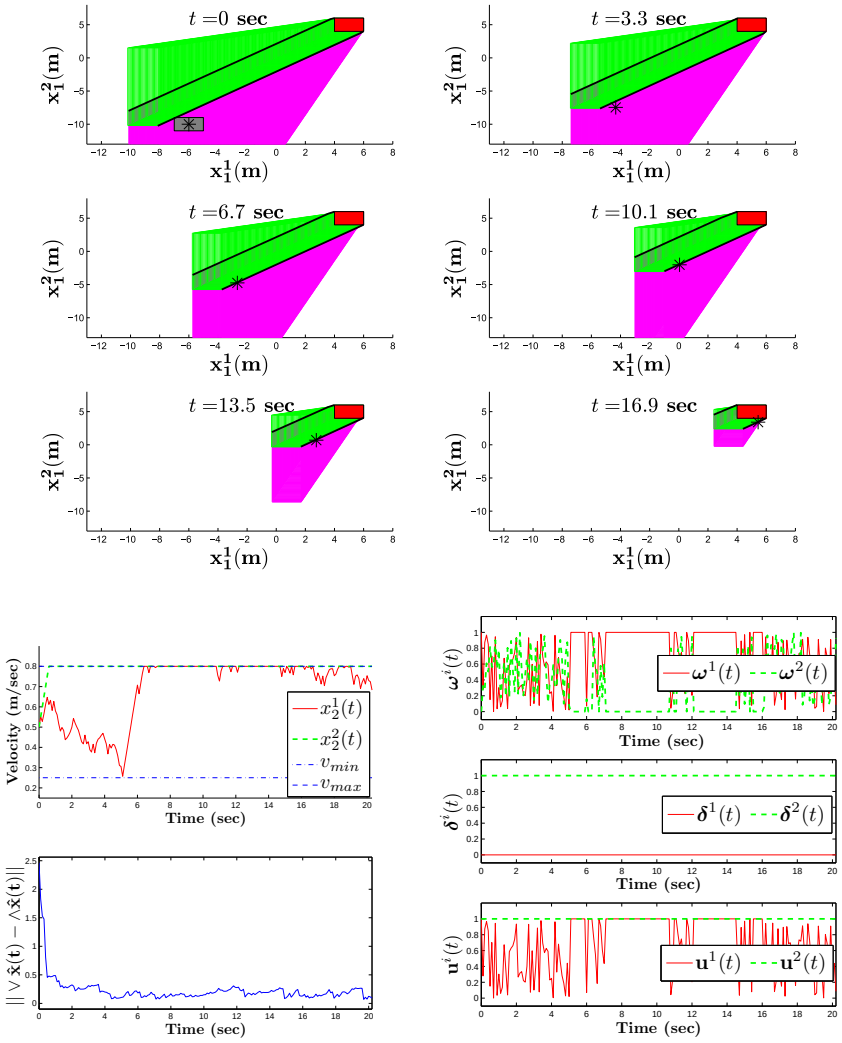


Figure 2.10: The competitive case. The above plots depict snapshots of the dynamic evolution of the closed-loop system. The system considered has $a^i = 1$ and $b^i = -.5$ for $i \in \{1,2\}$, with $v_{min} = .25$m/sec and $v_{max} = .8$m/sec. We choose $\Delta T = .1$ sec, $\mathbf{B} =]4,6[\times\mathbb{R}\times]4,6[\times\mathbb{R}$, $\Omega = [0,1]\times[0,1]$, $\Delta = [0,1]\times[0,1]$, $x_0 = (-6,.5,-10,.5)$, $\hat{x}_0 = [-7,-5]\times[.4,.6]\times[-11,-12]\times[.4,.6]$. The measurements z are generated randomly with a uniform probability distribution in the interval $[x(t) - (1,.1,1,.1), x(t) + (1,.1,1,.1)]$ so that $h(z) = [z-(1,.1,1,.1), z+(1,.1,1,.1)]$. The grey box represents the projection of $\hat{x}(t)$ onto the (x_1^1, x_1^2) plane, with the black asterisk representing the state of the system projected onto the (x_1^1, x_1^2) plane. The red box represents the projection of $\mathbf{B}$ onto the (x_1^1, x_1^2) plane, the slice of $C_{\omega_{\mathcal{L}}}$ corresponding to the current speeds is shown in green and the slice of $C_{\omega_{\mathcal{H}}}$ corresponding to the current speeds is shown in purple. Plots of the velocities, controls, disturbances, estimation error $\| \wedge \hat{x} - \vee\hat{x}\|$, and inputs are depicted in the lower panels.

2.7.3 The Combined Case: Experimental Results

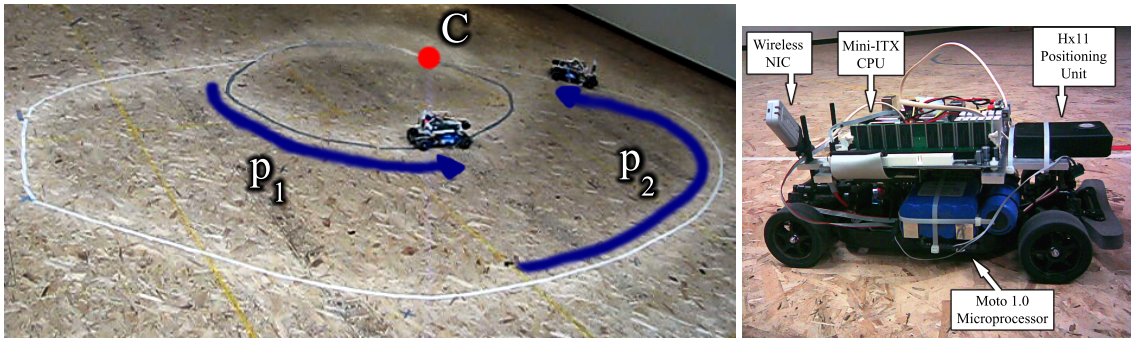


Figure 2.11: Roundabout test-bed (left). The longitudinal displacements of the vehicles with respect to a reference point along their corresponding paths are indicated by p_1 and p_2. The bad set **B** is a disk about point C. The vehicles (right).

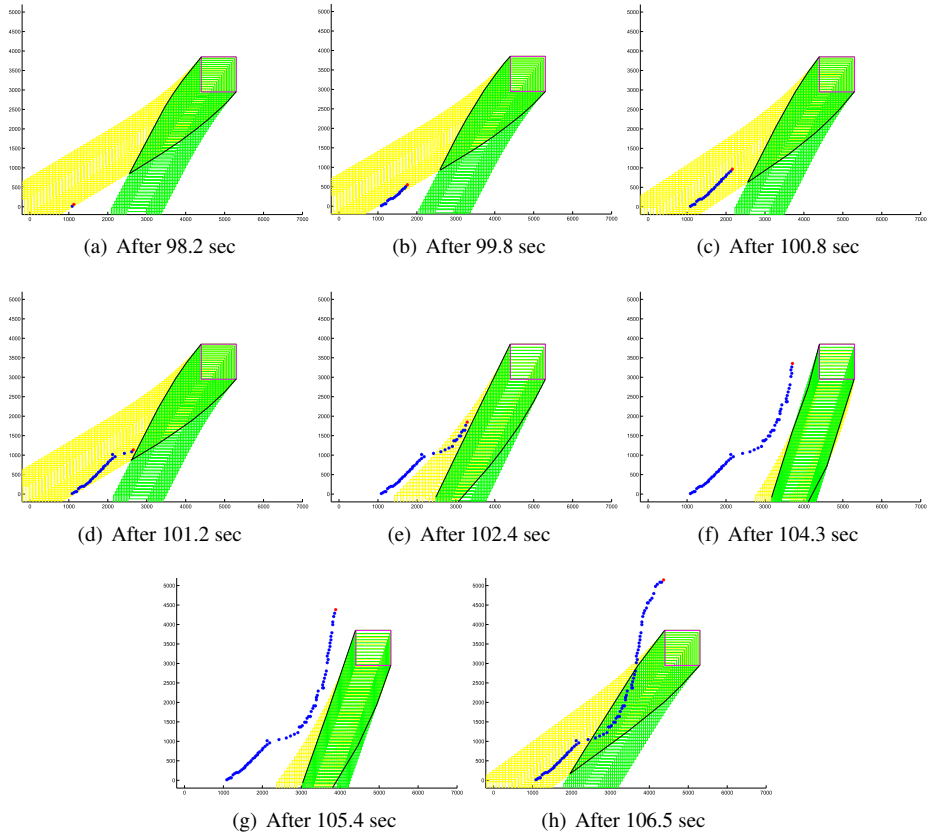
In order to show the suitability of the proposed algorithms for real-time applications, we implemented the algorithms on the in-scale roundabout test-bed shown in Figure 2.11. The vehicles are equipped with an on-board computer running Linux Fedora core, wireless (802.11b), speed and position sensors, and a motion controller that translates desired torque commands for the wheels into a PWM signal applied to the DC motor. This guarantees that the vehicle responds to torque commands (calculated in the on-board computer) through a second order dynamics of the type of equation (2.21). For a detailed description of the vehicles, the reader is referred to [97]. The dynamical parameters for each vehicle were experimentally determined and resulted in the longitudinal dynamics model $\ddot{p}_i = a^i\tau^i + b^i + \mathcal{D}^i = f_2^i((p_i, \dot{p}_i), \varphi(\tau^i, \mathcal{D}^i))$, in which $\tau^i \in [0, 100]$ is the percentage torque control command applied to the wheels from the motor, $a^1 = 1.20$ cm/sec^2, $b^1 = -0.90$cm/sec^2, $a^2 = 1.26$ cm/sec^2, $b^2 = -1.15$ cm/sec^2, $\mathcal{D}^1 \in [0.6, 19.1]$ cm/sec^2, and $\mathcal{D}^2 \in [0.85, 24.85]$ cm/sec^2. A torque command of 100% corresponds to a torque of 0.09 Nm. The terms $\mathcal{D}^i$ incorporate uncertainty that has been added to the model to take into account the parameter identification error. The limits on the speeds are taken as $v_{max} = 80$ cm/sec and $v_{min} = 25$ cm/sec. The speeds v_{max} and v_{min} given in the guard con-

ditions in Figure 2.8 are maintained through the employment of a proportional derivative (PD) speed control. The longitudinal dynamics model corresponds to a game model $\mathscr{G}$ in which $u^i = \varphi^i(\omega^i, \delta^i) = (\frac{\omega^1+\delta^1}{2}, \frac{\omega^2+\delta^2}{2})$ with $\omega^i \in [0.0, 200.0]$, $\delta^1 \in [0.6, 19.1]$ $2/a^1$, and $\delta^2 \in [0.85, 24.85]$ $2/a^2$.

Vehicle control has two main components: maintaining the vehicles on the corresponding roundabout paths and applying the appropriate control torques ω to the longitudinal dynamics to prevent collisions at point C (Figure 2.11). In general, the longitudinal and lateral dynamics of a vehicle are coupled. However, since the radii of the paths are much greater than the length of the vehicles and the speeds are low, it is possible to assume low coupling. This allows us to decouple the path following task, using a steering control input, from the longitudinal dynamics control, using the torque control input ω.

When no special torque command is required to guarantee safety (the last case of the control map in Theorem 4), a cruise control algorithm comes into effect to maintain the vehicle speeds about pre-defined set points. For the roundabout implementation, vehicle 1 tracks a speed of 0.4 m/s, while vehicle 2 tracks a speed of 0.5 m/s. A PD controller is employed for this tracking task. These speeds were selected such that the vehicles would be able to accelerate and decelerate as much as possible while staying in the speed range enforced by the speed limiter. The range of speeds was selected based on the geometry of the roundabout such that the capture set C does not extend beyond the reference point on either path. If this were not the case, the vehicles may apply control to avoid the bad set on the first pass, only to end up in the capture set for the second pass, thus making it impossible to avoid a collision.

Figure 2.12 illustrates the trajectory of the vehicle configuration projected onto the position plane, when avoiding a collision in one instance of the collision avoidance algorithm. The sets $C_{\omega_{\mathcal{L}}}$ and $C_{\omega_{\mathcal{H}}}$ are four dimensional. In the figure, we show the slices of these sets



(a) After 98.2 sec (b) After 99.8 sec (c) After 100.8 sec

(d) After 101.2 sec (e) After 102.4 sec (f) After 104.3 sec

(g) After 105.4 sec (h) After 106.5 sec

Figure 2.12: Experiment data showing the trajectory in the position plane of the vehicles configuration as it approaches a potential collision scenario. The red box is the projection of **B** in the position plane. In each panel, the green set represents a slice of the four dimensional set $C_{\omega_{\mathcal{H}}}$ corresponding to the current vehicles speeds. The yellow set represents a slice of the four dimensional set $C_{\omega_{\mathcal{L}}}$ corresponding to the current vehicles speeds. The red dot indicates the current vehicles positions. Control is applied at (d) to avoid the capture set, and the vehicles resume normal operation after passing the bad set (in (g) and (h)). The capture set slices are updated at every iteration on the basis of the vehicles speeds.

in the position plane corresponding to the current speeds of the vehicles.

CHAPTER III

Multi-Agent Testbed Implementation

In this chapter, we show how to design a provably safe robotic roundabout system by combining a number of control primitives. In particular, for the two-vehicle collision avoidance primitive, we exploit the natural partial order structure on which the system evolves to apply control techniques on partial orders, which have been shown to be computationally light [36, 49, 51]. We consider bounded state uncertainty derived from sensor noise and communication delay, and construct a state estimator. We show how to design the system variables in order to prevent conflicts among the control primitives and to thus ensure the safety and liveness of the system as a whole. We implement our design on a multi-vehicle hardware test-bed involving three vehicles continuously running on three different conflicting roundabouts. The experimental results show that the system is collision free and live.

3.1 Introduction

In this work, we solve the three-vehicle collision avoidance problem for a specific conflict topology. This problem is motivated by single lane modern roundabouts with multiple access points. Roundabouts are becoming common due to their impact on fuel economy and added safety benefits [7, 93]. To overcome the previously mentioned limitations in complexity, we exploit the fact that traffic systems can be well modeled by order pre-

serving systems. Our previous work involving two-vehicle collision avoidance exploited this structure to produce control primitives guaranteeing safety using linear complexity algorithms [36, 49, 96]. We now look to extend these methods to the three-vehicle safety control problem in which one vehicle can have simultaneous collision instances with the other two, see Figure 3.1. This is of practical significance in a roundabout where vehicles must avoid collisions with more than one vehicle merging into the roundabout. Central to our approach is the notion of modularity, which allows us to avoid computational difficulties inherent to the safety control problem for more than two-vehicles at a time. We accomplish this by combining control primitives (modules) through a conjunctive architecture where each module only guarantees part of the safety specification is met. We then show that this can be performed in a dead-lock free manner, thereby guaranteeing the whole system is safe.

The prior tools cited were developed for maintaining safety between two agents, thus we look to extend their definitions to guarantee safety for the complete three agent system. This is accomplished by modularizing the control maps that guarantee safety for the individual safety specifications outlined in Section 3.3. This chapter is organized as follows. In Section 3.2, we provide a description of the roundabout system, the models used for each vehicle. In Section 3.3, a formal description of the safety specification is given. A formal solution to this problem along with the feedback maps are provided in Section 3.4. The algorithms used to implement these feedback maps are provided in Section 3.5. In Section 3.6, we provide the experimental setup, the on-line control algorithm, and our experimental results.

3.2 System Model

We describe the system by introducing coordinates along each path, with the origins indicated as ticks along the paths in Figure 3.1. Assuming that each vehicle is constrained to evolve along the fixed path, we can completely describe the motion of each vehicle with the longitudinal dynamics along each path.

3.2.1 Longitudinal Dynamics

We denote by p the displacement of the vehicle along its path, and by v the velocity of the vehicle . The longitudinal dynamics along the vehicle path can be written as

$$\ddot{p} = [\mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)](f_w - f_{brake} - \frac{\rho_{air}}{2} C_D A_f (\dot{p})^2 - C_{rr}\mathcal{M}g),$$

in which $\mathcal{R}$ is the tire radius, J_w is the wheel inertia, $\mathcal{M}$ is the mass of the vehicle, $f_w = \tau_w^i \mathcal{R}$ where τ_w is the drive shaft output torque, f_{brake} is the brake force, ρ_{air} is the air density, C_D is the drag coefficient, A_f is the frontal area of the vehicle, C_{rr} is the rolling resistance, and g is the gravitational constant. For more details of this model, the reader is referred to [97] and the references therein. Assuming that the air drag is negligible, we can re-write the longitudinal dynamics as

$$\ddot{p} = au + b, \text{ where } u = f_w - f_{brake}, \; a = \frac{\mathcal{R}^2}{J_w + \mathcal{M}\mathcal{R}^2} \text{ and } b = -\frac{\mathcal{R}^2}{J_w + \mathcal{M}\mathcal{R}^2} C_{rr}\mathcal{M}g.$$

3.2.2 Piecewise Continuous Model

We next introduce a system model that incorporates a class of hybrid systems used to impose invariants on the velocity state. This system, denoted Σ^i, will be employed to model each vehicle $i \in \{1, 2, 3\}$.

Definition 9. *(Piecewise Continuous System)* A piecewise continuous system Σ^i is a tuple $\Sigma^i = (X^i, U^i, f^i, O^i)$, in which

(*i*) $X^i \subset R^2$ is a set of continuous states;

(*ii*) $U^i \subset R$ is a set of continuous inputs;

(*iii*) $O^i \subset X^i$ is a set of continuous outputs.

In particular, for each vehicle $i \in \{1, 2, 3\}$ the state space is given by $X^i = [0, D^i) \times [v^i_{min}, v^i_{max}]$, where D^i is the distance of the circular path as seen in Figure 3.1 and $0 < v^i_{min} < v^i_{max}$. For notation, let $X^i_1 = [0, D^i]$ and $X^i_2 = [v^i_{min}, v^i_{max}]$. When a vehicle traverses the complete path, that is $x^i_1(t) = D^i$, the displacement is set to $x^i_1(t) = 0$, thus X^i_1 is homeomorphic the circle $\mathbb{S}^1$. We assume $v^i_{min} > 0$ to enforce a liveness condition and a maximum velocity v^i_{max} to model the maximum speed limit. Let $x^i_j(t)$ denote the i^{th} agent and j^{th} component, where $j = 1$ corresponds to position and $j = 2$ corresponds to velocity. Define the set of all Lebesgue measurable functions $\mathbf{u}^i(\cdot) : \mathbb{R}_+ \to U^i$ that are essentially bounded as $\mathcal{U}^i$. The piecewise continuous vector field is represented as a hybrid automaton in Figure 3.2, where a^i and b^i are the vehicle parameters discussed in Section 3.2.1.

For vehicle $i \in \{1, 2, 3\}$, we denote the flow $\phi^i : \mathbb{R}_+ \times X^i \times \mathcal{U}^i \to X^i$, where for the initial condition $x^i \in X$ and input $\mathbf{u}^i \in \mathcal{U}^i$, we have $\phi(0, x^i, \mathbf{u}^i) = x^i$ and $\frac{d}{dt}\phi(t, x^i, \mathbf{u}^i) = f(\phi(t, x^i, \mathbf{u}^i), \mathbf{u}^i(t))$.

3.2.3 Parallel Composition

To formally describe the entire three vehicle roundabout system, we define the entire system as the parallel composition of each system, that is $\Sigma := \Sigma^1 || \Sigma^2 || \Sigma^3$. We next formally define the parallel composition among systems.

Definition 10. *(Parallel Composition)* For $\Sigma^i = (X^i, U^i, f^i)$ with $i \in \{1, 2, 3\}$, we define the *parallel composition* $\Sigma = \Sigma^1 || \Sigma^2 || \Sigma^3 := (X, U, f)$, in which $X := X^1 \times X^2 \times X^3$, $U := U^1 \times U^2 \times U^3$ and $f := (f^1, f^2, f^3)$.

For the initial condition $x = (x^1, x^2, x^3) \in X^1 \times X^2 \times X^3$, input $\mathbf{u} = (\mathbf{u}^1, \mathbf{u}^2, \mathbf{u}^3) \in$

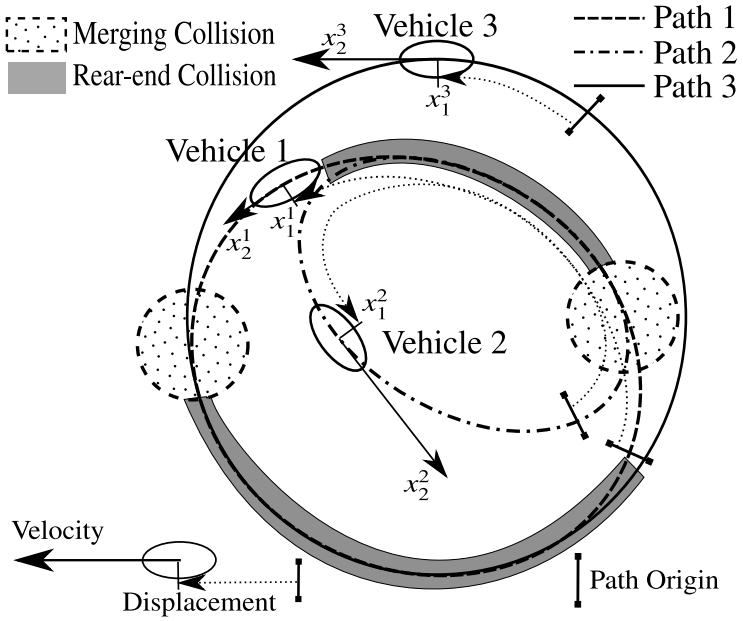


Figure 3.1: Roundabout system with locations of potential collisions superimposed on the paths, and vehicle states shown along with path origins.

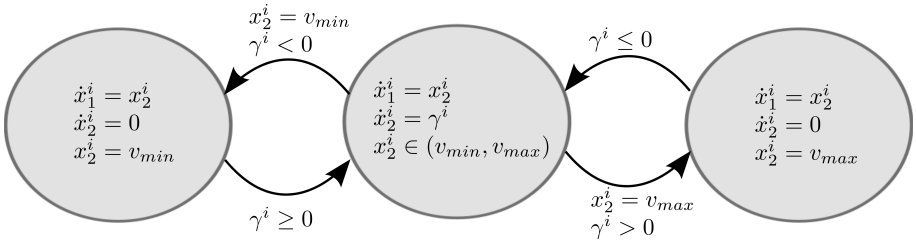


Figure 3.2: Hybrid system modeling the vehicle system Σ^i for $i \in \{1, 2, 3\}$. In the diagram, we have defined $\gamma^i := a^i u^i + b^i$.

$\mathcal{U}^1 \times \mathcal{U}^2 \times \mathcal{U}^3$, we denote the flow of the entire system $\Sigma = \Sigma^1 || \Sigma^2 || \Sigma^3$ as

$$\phi(t, x, \mathbf{u}) = (\phi^1(t, x^1, \mathbf{u}^1), \phi^2(t, x^2, \mathbf{u}^2), \phi^3(t, x^3, \mathbf{u}^3)),$$

where $\phi^1(t, x^1, \mathbf{u}^1) \in X^1$, $\phi^2(t, x^2, \mathbf{u}^2) \in X^2$ and $\phi^3(t, x^3, \mathbf{u}^3) \in X^3$.

In the sequel, we will frequently consider the state space and flow for only two vehicles. We introduce the vehicle collision index set, denoted by $\mathscr{I} \subset \mathbb{N}^2$, which represents all non-ordered pairs corresponding to vehicles for which collision is possible. For the system generated by the roundabout in Figure 3.1, the vehicle collision index set is given as

$$\mathscr{I} := \{(1, 2), (1, 3)\}. \tag{3.1}$$

For any collision index $k \in \mathscr{I}$, we use the following notation. Define the two vehicle state space by $X^k := X^{k_1} \times X^{k_2}$ and a component of the state space by $X^k_j := X^{k_1}_j \times X^{k_1}_j$. Similarly, define the state vector $x^k := (x^i, x^j)$ and a component of the state vector by $x^k_j := (x^{k_1}_j, x^{k_2}_j)$. The two-vehicle input space is defined by $U^k := U^{k_1} \times U^{k_2}$, with the input signal defined by $\mathbf{u}^k := (\mathbf{u}^{k_1}, \mathbf{u}^{k_2})$. The two-vehicle flow is defined by $\phi^k(t, x^k, \mathbf{u}^k) := (\phi^{k_1}(t, x^{k_1}, \mathbf{u}^{k_1}), \phi^{k_2}(t, x^{k_2}, \mathbf{u}^{k_2}))$ and a component of the flow defined by $\phi^k_j(t, x^k, \mathbf{u}^k) := (\phi^{k_1}_j(t, x^{k_1}, \mathbf{u}^{k_1}), \phi^{k_2}_j(t, x^{k_2}, \mathbf{u}^{k_2}))$.

3.3 Problem Statement

To formalize the safety control problem, we introduce a set $\mathbf{B} \subset X$, called the *bad set*, representing system configurations that violate a safety condition for the *entire* system Σ. We will say that the set $\mathbf{B}$ represents a global safety specification. We construct this set by taking the union of all collision configurations between the vehicles.

From the geometry of the roundabout, we can classify all collisions into two types (Figure 3.1). We call the first type a *merging collision*, denoted by the specification M, which occurs when vehicles collide as their paths first come together, an example is provided in

Figure 3.3. We call the second type a *rear-end collision*, denoted by the specification RE, which occurs when vehicles collide while their paths overlap, an example is provided in Figure 3.4.

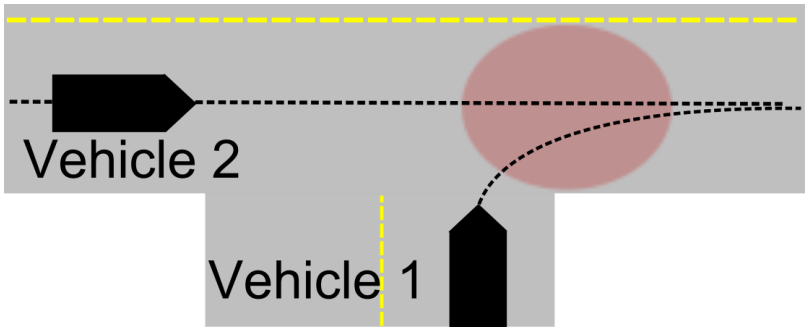


Figure 3.3: Example of merging collision scenario at an intersection. It is assumed that a merging collision occurs when both vehicles simultaneously lie within the red circle.

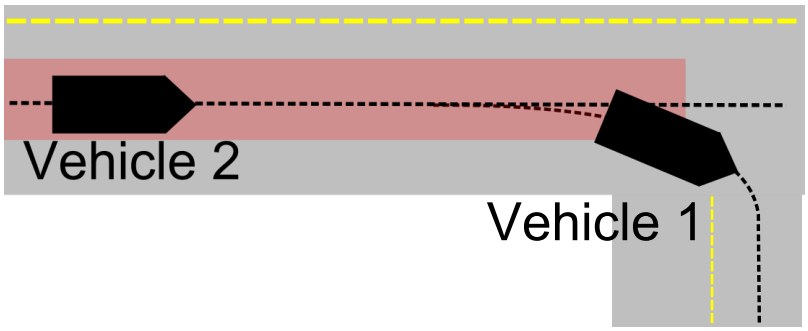


Figure 3.4: Example of rear-end collision scenario along overlapping vehicle paths. It is assumed that a rear-end collision is only possible within the red box.

3.3.1 Construction of the Bad Set

Define the two-vehicle bad set $B_s^k \subset X^k$ for a collision type $s \in \{M, RE\}$ between vehicles $k \in \mathscr{I}$. If two vehicles are close enough to each other, then a collision occurs regardless of velocity. Therefore, instances of collision are determined solely by vehicle positions. Let $\Omega_s^k \subset X_1^k$ be the set of all positions that generate a collision, then we define

$$B_s^k := \{x^k \in X^k \mid x_1^k \in \Omega_s^k\}.$$

We will call the space X_1^k the *space of constraint* for the two-vehicle bad set B_s^k. The two-vehicle bad set can be used to define a *local* safety specification.

Definition 11. *(Local Safety Specification)* Given the bad set $B_s^k \subset X^k$, initial condition $x^k \in X^k$ and input $\mathbf{u}^k \in \mathcal{U}^k$, we say the *local* safety specification is met if $\phi^k(t, x^k, \mathbf{u}^k) \notin B_s^k$ for all $t \in \mathbb{R}_+$.

We call this safety specification, since it pertains only to a specific two-vehicle collision.

3.3.2 Collisions Considered

We now mathematically construct all instances of two-vehicle bad sets that are possible within the roundabout system (Figure 3.1).

Merging Collisions

From the path geometries, there only exist merging collisions between vehicles of index $k \in \mathscr{I}$ (defined in (3.1)), as shown by the circle sets in Figure 3.1. For $k \in \mathscr{I}$, define the set Ω_M^k corresponding to positions that generate merging collisions as

$$\Omega_M^k \quad := \quad \left\{ x_1^k \in X_1^k \mid x^k \in]L_M^k, U_M^k[\right\},$$

where $0 < L_M^k < U_M^k < \min\{D_1^k, D_2^k\}$ are positive constants. The set Ω_M^k represents the collection of all positions of vehicles k_1 and k_2 such that both lie in the open interval $]L_M^k, U_M^k[$. This set is shown for both $k_2 = 2$ and $k_2 = 3$ in Figure 3.6.

We use the set Ω_M^k to define the merging two-vehicle merging bad set as

$$B_M^k := \left\{ x^k \in X^k \mid x_1^k \in \Omega_M^k \right\}.$$

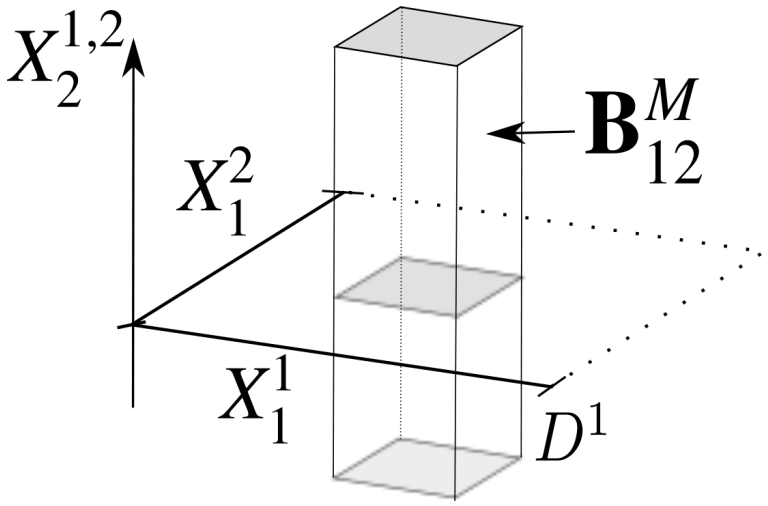


Figure 3.5: The two-vehicle bad set $B_M^{1,2}$ is depicted within $X^{1,2}$.

Rear-End Collisions

From the path geometries in Figure 3.1, there only exist rear-end collisions between vehicles $\{1, 2\}$ and between vehicles $\{1, 3\}$. We present the rear-end safety specification by first constructing the bad sets in the space of constraint. Define the avoid sets $A_{RE}^{(1,2)}$ and $A_{RE}^{(1,3)}$ corresponding to positions that generate rear-end collisions as

$$\Omega_{RE}^k := \left\{ x_1^k \in X_1^k \mid x_1^k \in]L_{RE}^k, U_{RE}^k[\ \wedge \ |x_1^{k_1} - x_1^{k_2}| < l \right\},$$

where $0 < L_{RE}^k, U_{RE}^k < \min\{D^{k_1}, D^{k_2}\}$ are positive constants, and l denotes a length of the vehicle. We note that $L_M^k = L_{RE}^k$ by construction. This set corresponds to sections of the roundabout where vehicle paths overlap, and each vehicle is a car length apart. This set is shown in Figure 3.6. We use the sets $\Omega_{RE}^{(1,2)}$ and $\Omega_{RE}^{(1,3)}$ to construct the rear-end bad sets as $B_{RE}^k := \{x^k \in X^k \mid x_1^k \in \Omega_{RE}^k\}$.

3.3.3 Complete Safety Specification

For the spec $s \in \{M, RE\}$, we extend the two-vehicle bad set $B_s^k \subset X^k$ to the bad set $\mathbf{B}_s^k \subset X$ for the entire system by defining $\mathbf{B}_s^k := \{x \in X \mid x^k \in B_s^k\}$. It naturally follows from the

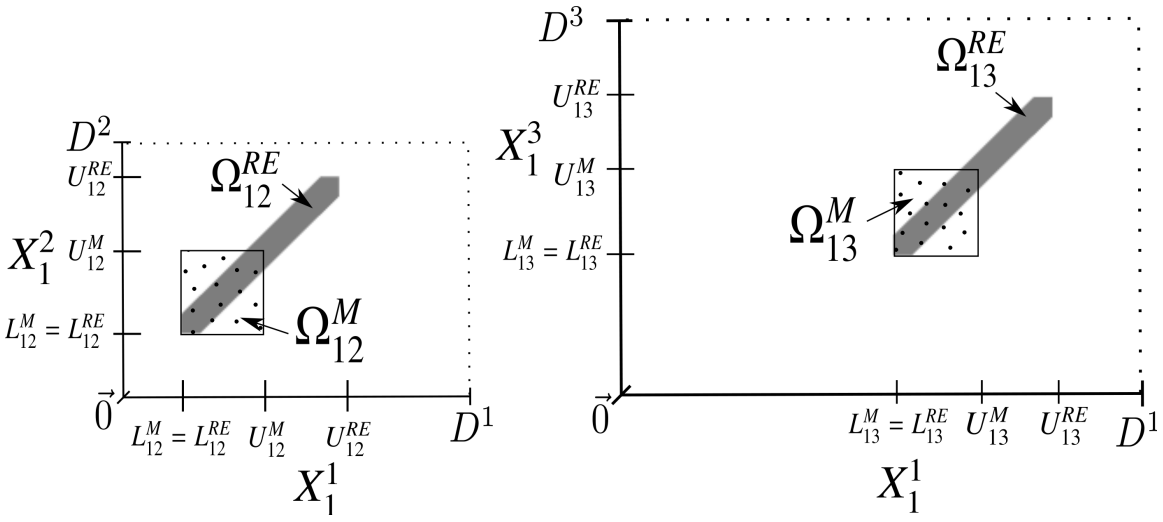


Figure 3.6: The sets $\Omega_M^{(1,2)}$, $\Omega_M^{(1,3)}$, $\Omega_{RE}^{(1,2)}$ and $\Omega_{RE}^{(1,3)}$ projected onto the relevant Spaces of Constraint.

above definition that $x^k \notin B_s^k$ if and only if $x \notin \mathbf{B}_s^k$ for all states $x \in X$. That is, the two-vehicle component of the state vector x^k is outside of the two-vehicle bad set B_s^k if and only if the entire state vector is outside of the bad set $\mathbf{B}_s^k$. The set $\mathbf{B}_M^k$ is shown in Figure 3.5.

We define the bad set as $\mathbf{B} := \mathbf{B}_M^{(1,2)} \cup \mathbf{B}_M^{(1,3)} \cup \mathbf{B}_{RE}^{(1,2)} \cup \mathbf{B}_{RE}^{(1,3)}$. With the definition of the bad set $\mathbf{B}$, we now define the *global* safety specification for the complete system.

Definition 12. *(Global Safety Specification)*

The system Σ, along with initial condition $x_0 \in X$ and input $\mathbf{u} \in \mathcal{U}$, is said to meet the *global* safety specification if $\phi(t, x_0, \mathbf{u}) \notin \mathbf{B}$ for all $t \in \mathbb{R}_+$.

3.3.4 Control Modules

For each two-vehicle bad set B_s^k, we define the two-vehicle *control primitive* as a feedback map into the two-vehicle input space U^k, used to maintain the local safety specification as defined in Definition 11. The primitive is then extended to a *module*, which is defined as a

feedback map into the entire Input space U.

Cooperative Control Primitive

We assume all merging collisions are to be resolved by both vehicles operating in a cooperative fashion. For the two-vehicle bad set B_M^k between $k \in \mathscr{I}$, we define the *capture* set C_M^k as

$$C_M^k := \{x^k \in X^k \mid \forall\ \mathbf{u}^k \in \mathcal{U}^k,\ \exists\ t \in \mathbb{R}_+ \text{ s.t. } \phi^k(t, x^k, \mathbf{u}^k) \in B_M^k\}.$$

That is, we seek to find the set of all initial conditions x^k in the space X^k such that for every input signal $\mathbf{u}^k \in \mathcal{U}^k$ applied to vehicles k_1 and k_1, the flow enters B_M^k at some time $t \in \mathbb{R}_+$.

From this set, we then define the cooperative control *primitive* g_M^k.

Definition 13. *(Cooperative Primitive)*

For the two-vehicle bad set B_M^k and the capture set C_M^k, define the cooperative control primitive $g_M^k(\cdot) : X^k \rightrightarrows U^k$ such that if $x^k \in X^k \backslash C_M^k$, then

$$\phi^k(t, x^k, \mathbf{u}^k) \notin C_M^k \quad \forall\ t \in \mathbb{R}_+,$$

under any feedback signal $\mathbf{u}^k(\tau) \in g_M^k(\phi^k(\tau, x^k, \mathbf{u}^k))$ for all $\tau \in [0, t]$.

The cooperative primitive acts in the least conservative manner possible, that is to say control is only applied when absolutely necessary.

Competitive Control Primitive

We next define a competitive two-vehicle control primitive, used to maintain the local safety specification for the two-vehicle bad set B_{RE}^k. For the competitive primitive, we only assume control authority for vehicle k_2. This primitive will be useful in guaranteeing the local safety specification while only requiring control of one vehicle. Two accommodate the worst case, the uncontrolled vehicle is modeled as a disturbance attempting to violate the local two-vehicle safety specification.

For the two-vehicle bad set B^k_{RE} between vehicles $k \in \mathscr{I}$, we define the competitive two-vehicle capture set C^k_{RE} as

$$C^k_{RE} := \{x^k \in X^k \mid \forall\ \mathbf{u}^{k_2} \in \mathcal{U}^{k_2},\ \exists\ \mathbf{u}^{k_1} \in \mathcal{U}^{k_1},\ \exists\ t \in \mathbb{R}_+ \text{ s.t. } \phi^k(t, x^k, \mathbf{u}^k) \in B^k_{RE}\}.$$

That is, we seek to find the set of initial conditions x^k in the two vehicle space X^k such that for every input signal $\mathbf{u}^{k_2} \in \mathcal{U}^{k_2}$ applied to vehicle k_2, there exists an input signal $\mathbf{u}^{k_1} \in \mathcal{U}^{k_1}$ such that the flow enters B^k_{RE} at some time $t \in \mathbb{R}_+$.

From this set, we then define the competitive control *primitive* g^k_{RE}.

Definition 14. *(Competitive Primitive)* For the two-vehicle bad set B^k_{RE} and the capture set C^k_{RE}, define the competitive control primitive $g^k_{RE}(\cdot) : X^k \rightrightarrows U^{k_2}$ such that if $x^k \in X^k \backslash C^k_{RE}$, then $\phi^k(t, x^k, \mathbf{u}^k) \notin C^k_{RE}$ for all $t \in \mathbb{R}_+$ under any input signal $\mathbf{u}^{k_1} \in \mathcal{U}^{k_1}$ and any feedback signal $\mathbf{u}^{k_2}(\tau) \in g^k_{RE}(\phi^k(\tau, x^k, \mathbf{u}^k))$ for all $\tau \in [0, t]$.

This control module is useful in that only one vehicle is tasked with maintaining the safety specification. In this sense, the behavior of this primitive is far more conservative than the cooperative primitive. This module is useful in cases where a single vehicle could be simultaneously engaged in two possible collision scenarios, and hence cannot use cooperative primitives for both collisions due to the possibility that they could command opposite control input at a single instant.

Control Modules from Primitives

The cooperative primitive only applies control to two-vehicles while the competitive primitive only applies control to one-vehicle, therefore we look to extend the definition of primitives to the entire input set U so they can be combined. In addition, we define an *enabling* set $\mathbf{A} \subset X$ used to turn the module on and off based on the input argument.

Given the cooperative primitive g^k_M between vehicles $k \in \mathscr{I}$, we first define the cooperative *module*.

Definition 15. *(Cooperative Module)*

Given cooperative primitive g_M^k and enable set $\mathbf{A}_M^k \subset X$ where we require that $\mathbf{B}_M^k \subset \mathbf{A}_M^k$, define the cooperative module $\mathbf{g}_M^k : X \rightrightarrows U$ as

$$\mathbf{g}_M^k(x) = \begin{cases} \{u \in U \mid u^k \in g_M^k(x^k)\} & \text{if } x \in \mathbf{A}_M^k, \\ U & \text{else.} \end{cases}$$

We say that $\mathbf{g}_M^k$ is the *least restrictive extension*, since any control input satisfying the lower dimensional constraint $g_M^k(x^k) \subset U^k$ is admissible when x is inside the enable set $\mathbf{A}_M^k \subset X$.

Given the competitive primitive g_{RE}^k between vehicles $k \in \mathscr{I}$, we next define the competitive *module*.

Definition 16. *(Competitive Module)*

Given competitive primitive g_{RE}^k and enable set $\mathbf{A}_{RE}^k \subset X$ where we require that $\mathbf{B}_{RE}^k \subset \mathbf{A}_{ij}^{RE}$, define the cooperative module $\mathbf{g}_{RE}^k : X \rightrightarrows U$ as

$$\mathbf{g}_{RE}^k(x) = \begin{cases} \{u \in U \mid u^j \in g_{RE}^k(x^k)\} & \text{if } x \in \mathbf{A}_{RE}^k, \\ U & \text{else.} \end{cases}$$

We say that $\mathbf{g}_{RE}^k$ is the *least restrictive extension*, since any control input satisfying the lower dimensional constraint $g_{RE}^k(x^k) \subset U^{k_2}$ is admissible when x is inside the enable set $\mathbf{A}_M^k \subset X$.

We next define the composition of two modules utilizing a conjunctive control architecture.

Definition 17. *(Composition of Modules)*

For the modules $\mathbf{g}_r^k(x)$ and $\mathbf{g}_s^l(x)$, where $k, l \in \mathscr{I}$ and $r, s \in \{M, RE\}$, define the composition $\mathbf{g} : X \rightrightarrows U$ as

$$\mathbf{g}(x) := \mathbf{g}_r^k(x) \cap \mathbf{g}_s^l(x).$$

It is not necessarily true that the resulting combination of two modules will maintain both local safety specifications (Definition 11). To establish conditions guaranteeing the combination of modules maintains both safety specifications, for $k \in \mathscr{I}$ and $s \in \{M, RE\}$ we first define the capture set $\mathbf{C}_s^k \subset X$ extended into the entire space X.

Definition 18. Given the two-vehicle capture set $C_s^k \subset X^k$ with $s \in \{M, RE\}$ and $k \in \mathscr{I}$, we define the capture set $\mathbf{C}_s^k := \{x \in X \mid x^k \in C_s^k\}$.

3.4 Problem Solution

The problem of constructing the cooperative control module (Definition 15) and the competitive control module (Definition 16) has already been solved in [36], [49]. In this work, the central assumption is that the system under consideration falls into the class of order preserving systems. We first introduce this abstraction.

3.4.1 Order Preserving Systems

Order preserving systems comprise a class of dynamical systems that are useful in describing vehicles commonly encountered in traffic systems. Before introducing the specifics, we first introduce the notion of a partial order.

A partial order is a set P along with a relation $\leq$, denoted $(P, \leq)$. We are concerned with the partial order $(\mathbb{R}^n, \leq)$ defined by component-wise ordering, and the partial order $(\mathcal{U}, \leq)$, where we say $\mathbf{u}_1 \leq \mathbf{u}_2$ given that $\mathbf{u}_1(t) \leq \mathbf{u}_2(t)$ for all $t \in \mathbb{R}_+$.

We next define an order preserving map. Given partial orders $(P, \leq_p)$, $(S, \leq_s)$ and a map $F : P \to S$, we say that F is *order preserving* if $x \leq_p y$ implies $F(x) \leq_s F(y)$. The definition of an order preserving map allows us to formally define the class of *order preserving* systems.

Definition 19. *(Order Preserving System)*

The system $\Sigma^i = (X^i, U^i, f^i)$, for $i \in \{1, 2, 3\}$, is an Order Preserving System provided that there exist constants $u_L^i, u_H^i \in \mathbb{R}$ and $\chi^i > 0$ such that

(i) $U^i = [u_L^i, u_H^i] \subset \mathbb{R}$;

(ii) The flow $\phi^i(t, x^i, \mathbf{u}^i)$ generated by the vector field f is an order preserving map with respect to $(X, \leq)$ and $(\mathcal{U}, \leq)$;

(iii) $f_1^i(x, u) > \chi^i$ for all $(x, u) \in X^i \times U^i$.

Condition (i) states that the input to vehicle $i \in \{1, 2, 3\}$ falls within the bounded interval $[u_L^i, u_H^i]$. Condition (ii) with respect to state implies that greater initial speeds and initial displacements generate larger speeds and displacements. Condition (ii) with respect to input implies that greater inputs generate greater speeds and displacements. Mathematically for vehicle $i \in \{1, 2, 3\}$, this is stated for $\tilde{x}^i, \tilde{y}^i \in \tilde{X}^i$ and $\mathbf{u}^i, \hat{\mathbf{u}}^i \in \mathcal{U}$ as

$$x^i \leq y^i,\ \mathbf{u}^i \leq \hat{\mathbf{u}}^i \Rightarrow \phi^i(t, x^i, \mathbf{u}^i) \leq \phi^i(t, y^i, \hat{\mathbf{u}}^i) \text{ for all } t \in \mathbb{R}_+.$$

Condition (iii) guarantees that a liveliness condition is always maintained.

3.4.2 Primitive Generation

For the two-vehicle index $k \in \mathscr{I}$ and a system $\Sigma := \Sigma^{k_1} \| \Sigma^{k_2}$ defined as the parallel composition of two order preserving systems, and an order preserving connected (o.p.c.) bad set $B^k \subset X^k := X^{k_1} \times X^{k_2}$ [49], we construct the control primitives (Definitions 13, 14).

For the cooperative control primitive, all results are taken from [36]. The set $C^k(\mathbf{u}^k)$ corresponds to the set of all states that are taken into B^k under the fixed input $\mathbf{u}^k \in \mathcal{U}^k$. We present the control primitive generated from the restricted cooperative capture set.

Theorem 5. *(Cooperative Primitive Construction)*

For the two-vehicle index $k \in \mathscr{I}$, order preserving system $\Sigma := \Sigma^{k_1} \| \Sigma^{k_2}$ and the o.p.c.

bad set $B^k \subset X^k$, the cooperative primitive $g^k : X^k \rightrightarrows U^k$ is given by

$$g^k(x^k) := \begin{cases} u^k_{\mathcal{L}} & \text{if } x^k \in \partial C^k(\mathbf{u}^k_{\mathcal{L}}) \cap C^k(\mathbf{u}^k_{\mathcal{H}}), \\ u^k_{\mathcal{H}} & \text{if } x^k \in \partial C^k(\mathbf{u}^k_{\mathcal{H}}) \cap C^k(\mathbf{u}^k_{\mathcal{L}}), \\ \{u^k_{\mathcal{H}}, u^k_{\mathcal{L}}\} & \text{if } x^k \in \partial C^k(\mathbf{u}^k_{\mathcal{L}}) \cap \partial C^k(\mathbf{u}^k_{\mathcal{H}}), \\ U^k & \text{else,} \end{cases}$$

where $u^k_{\mathcal{H}} := (u^{k_1}_L, u^{k_2}_H)$, $u^k_{\mathcal{L}} := (u^{k_1}_L, u^{k_2}_H)$, the constant signal $\mathbf{u}^k_{\mathcal{H}} = u^k_{\mathcal{H}}$ for all $t \in \mathbb{R}_+$ and the constant signal $\mathbf{u}^k_{\mathcal{L}} = u^k_{\mathcal{L}}$ for all $t \in \mathbb{R}_+$.

This primitive has been shown to render the set $X \backslash C^k$ controlled invariant, where it has been shown that the cooperative capture set can be found as $C^k = C^k(\mathbf{u}^k_{\mathcal{L}}) \cap C^k(\mathbf{u}^k_{\mathcal{H}})$.

For the competitive control primitive, results are taken from [49]. The set $C^k(\mathbf{u}^{k_2})$ corresponds to the set of all states that are taken into B^k under the fixed input $\mathbf{u}^{k_2} \in \mathcal{U}^{k_2}$ and any input $\mathbf{u}^{k_1} \in \mathcal{U}^{k_1}$. We present the competitive control primitive generated with the restricted competitive capture sets.

Theorem 6. *(Competitive Primitive Construction)*

For the two vehicle index $k \in \mathscr{I}$ and order preserving system $\Sigma := \Sigma^{k_1} \| \Sigma^{k_2}$ and the o.p.c. bad set $B^k \subset X^k$, the competitive primitive $g^k : X^k \rightrightarrows U^{k_2}$ is given by

$$g^k(x^k) := \begin{cases} u^{k_2}_L & \text{if } x^k \in \partial C^k(\mathbf{u}^{k_2}_L) \cap C^k(\mathbf{u}^{k_2}_H), \\ u^{k_2}_H & \text{if } x^k \in \partial C^k(\mathbf{u}^{k_2}_H) \cap C^k(\mathbf{u}^{k_2}_L), \\ \{u^{k_2}_H, u^{k_2}_L\} & \text{if } x^k \in \partial C^k(\mathbf{u}^{k_2}_L) \cap \partial C^k(\mathbf{u}^{k_2}_H), \\ U^2 & \text{else,} \end{cases}$$

where $\mathbf{u}^{k_2}_H := u^{k_2}_H$ for all $t \in \mathbb{R}_+$ and $\mathbf{u}^{k_2}_L := u^{k_2}_L$ for all $t \in \mathbb{R}_+$.

This primitive has been shown to render the set $X \backslash C_{12}$ controlled invariant for any input signal $\mathbf{u}^1 \in \mathcal{U}^1$, where it has been shown that the competitive capture set can be found as $C^k = C^k(\mathbf{u}^{k_2}_L) \cap C^k(\mathbf{u}^{k_2}_L)$.

3.4.3 Extended System Description

The problem with the system Σ^i, as defined in Section 3.2.2 for $i \in \{1, 2, 3\}$, is that one cannot place the natural component-wise partial order on the set X_1^i, as it is homeomorphic to $\mathbb{S}^1$. Therefore, we extend the state space to allow for a partial order.

The Covering Space $\tilde{X}$

We introduce the *covering space* $\tilde{X}^i$, where $\tilde{X}_1^i := \mathbb{R}$ and $\tilde{X}_2^i := X_2^i$, along with the *covering map* $\xi^i : \tilde{X}^i \to X^i$ [74]. For a state $\tilde{x}^i \in \tilde{X}^i$, the covering map ξ^i is defined by

$$x^i =: \xi^i(\tilde{x}^i) \text{ where } x_2^i = \tilde{x}_2^i \text{ and } x_1^i = \tilde{x}_1^i + kD^1 \text{ for some } k \in \mathbb{Z} \text{ and } x_1^i \in X_1^i.$$

The notation ξ_j^i for $j \in \{1, 2\}$ corresponds to components of the image of the covering map ξ^i.

The covering space corresponds to viewing the position of vehicle $i \in \{1, 2, 3\}$ on the real line instead of on the circle X_1^i. If a vehicle $i \in \{1, 2, 3\}$ traveled around the path multiple times, the state $\tilde{x}_1^i \in \tilde{X}_1^i$ would reflect this, while the state $x_1^i \in X_1^i$ only defines the directed distance from the path origin. For an element $x^i \in X^i$, denote the *equivalence class* $[x^i] \subset \tilde{X}_1^i$ as

$$[x^i] := \{\tilde{x}^i \in \tilde{X} \mid \xi^i(\tilde{x}^i) = x^i\}.$$

We say that two elements $\tilde{x}, \tilde{y} \in \tilde{X}_1^i$ are equivalent, denoted $\tilde{x} \sim \tilde{y}$, provided that $\tilde{x}, \tilde{y} \in [x_1^i]$. The position space X_1^i, covering space $\tilde{X}_1^i$ and covering map ξ_1^i are each depicted in Figure 3.7.

In the space $\tilde{X}^i$, we can define the partial order $(\tilde{X}^i, \leq)$ using the standard component-wise ordering.

We introduce an extension operator that allows us to take a system Σ^i (as defined in Section 3.2.2) and map it to a system $\tilde{\Sigma}^i$ with a state space defined to be the covering

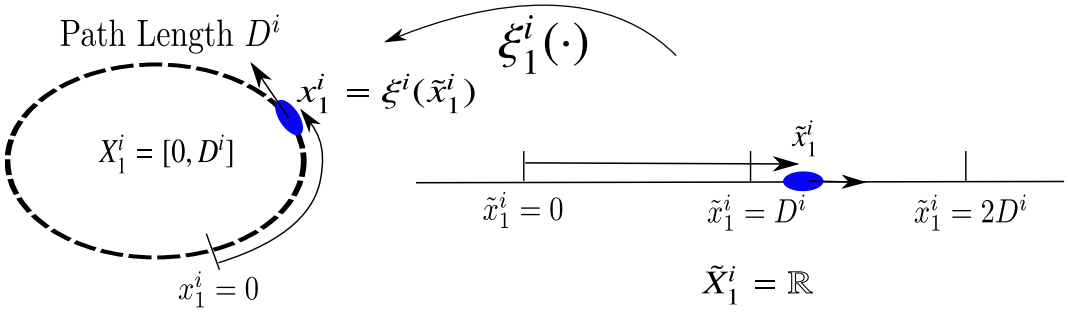


Figure 3.7: The Position Space X_1^i, Covering Space $\tilde{X}_1^i$ and Covering Map ξ^i for Vehicle $i \in \{1, 2, 3\}$.

space.

Definition 20. *(Extended System Operator)* Given the system Σ^i for vehicle $i\{1, 2, 3\}$, define the extended system $\tilde{\Sigma}^i$ and operator Υ where $\tilde{\Sigma}^i := \Upsilon(\Sigma^i)$, such that

$$\tilde{\Sigma}^i = \{\tilde{X}^i, U^i, \tilde{f}^i\},$$

where the vector field $\tilde{f}^i : \tilde{X}^i \times U^i \to \tilde{X}^i$ is such that

$$\xi^i(\tilde{f}^i(\tilde{x}^i, u^i)) = f(\xi^i(\tilde{x}^i), u^i)$$

for all $\tilde{x}^i \in \tilde{X}^i$ and $u^i \in U^i$.

We define the extended flow $\tilde{\phi}$ generated by the extended system $\tilde{\Sigma}^i$. Let $\tilde{\phi} : \mathbb{R}_+ \times \tilde{X}^i \times \mathcal{U} \to \tilde{X}^i$, where for an initial condition $\tilde{x}^i \in \tilde{X}^i$ and input $\mathbf{u}^i \in \mathcal{U}^i$, we have that $\tilde{\phi}(0, \tilde{x}^i, \mathbf{u}^i) = \tilde{x}^i$ and

$$\xi^i(\tilde{\phi}^i(t, \tilde{x}^i, \mathbf{u}^i)) = \phi^i(t, \xi^i(\tilde{x}^i), \mathbf{u}^i).$$

We next provide a result that shows that extended system generated by the extension of each piecewise continuous system is of the order preserving systems class.

Remark 1. *Given the piecewise continuous system Σ^i where $i \in \{1, 2, 3\}$ (Definition 9), the system $\tilde{\Sigma}^i$ generated by extension operator $\Upsilon(\Sigma^i)$ an order preserving system.*

Proof. Condition (i) holds trivially from the definition of U^i. □

(*i*) $U^i = [u^i_L, u^i_H] \subset \mathbb{R}$;

(*ii*) The flow $\phi^i(t, x^i, \mathbf{u}^i)$ generated by the vector field f is an order preserving map with respect to $(X, \leq)$ and $(\mathcal{U}, \leq)$;

(*iii*) $f^i_1(x, u) > \chi^i$ for all $(x, u) \in X^i \times U^i$.

Parallel Composition

We can extend this definition to the parallel composition of extended systems, that is $\tilde{\Sigma} := \tilde{\Sigma}^1 || \tilde{\Sigma}^2 || \tilde{\Sigma}^3$. All of the above notation regarding the flow ϕ of the entire system Σ carries over to the flow $\tilde{\phi}$ of the extended entire system $\tilde{\Sigma}$. In particular, we have that $\tilde{\phi} : \mathbb{R}_+ \times \tilde{X} \times \mathcal{U} \to \tilde{X}$ is order preserving with respect to $(\tilde{X}, \leq)$ and $(\mathcal{U}, \leq)$. We also can define the covering map ξ defined on the entire system $\tilde{\Sigma}$, given by $\xi(\tilde{x}) = (\xi^1(\tilde{x}^1), \xi^2(\tilde{x}^2), \xi^3(\tilde{x}^3))$

All of the notation pertaining to the two-vehicle description, as in Section 3.2.3, carries over naturally to the extended system $\tilde{\Sigma}$.

3.4.4 Extended Safety Specification

For each bad set $B^k_s \subset X^k$ generated by the safety specification $s \in \{M, RE\}$ for two-vehicle index $k \in \mathscr{I}$, we can extend the bad set into the covering space $\tilde{X}^k$. Define the extended bad set $\tilde{B}^k_s \subset \tilde{X}^k$ generated by the B^k_s as

$$\tilde{B}^k_s := \{\tilde{x}^k \in \tilde{X}^k \mid \xi^k(\tilde{x}^k) \in B^k_s\}.$$

This set can be further embedded into the entire extended statespace as $\tilde{\mathbf{B}}^k_s = \{\tilde{x} \in \tilde{X} \mid \tilde{x}^k \in B^k_s\}$. The bad set $\mathbf{B}$ will be denoted $\tilde{\mathbf{B}}$ in the covering space. The extended bad set will allow us to view the safety specification as a subset of a partially ordered set, which will be useful in invoking previous results concerning safety control for order preserving systems.

The safety specification can be described in terms of the extended bad set.

Condition 1. Given the system Σ, bad set $\tilde{\mathbf{B}}$, initial condition $x_0 \in X$ and input $\mathbf{u} \in \mathcal{U}$, we say the system is safe if the for any $\tilde{x}_0 \in \tilde{X}$ such that $\xi(\tilde{x}_0) = x_0$, the extended system $\Upsilon(\Sigma)$ generates a flow $\tilde{\phi}(t, \tilde{x}_0, \mathbf{u})$ never enters the extended bad set $\tilde{\mathbf{B}}$, that is $\tilde{\phi}(t, \tilde{x}_0, \mathbf{u}) \notin \tilde{\mathbf{B}} = \emptyset$ for all $t \in \mathbb{R}_+$.

This condition is equivalent to Condition 12 from Section 3.3.3, in that $\xi(\tilde{\mathbf{B}}) = \mathbf{B}$ and $\xi(\tilde{\phi}(t, \tilde{x}_0, \mathbf{u})) = \phi(t, x_0, \mathbf{u})$, however now we have described the condition within the partially ordered space $(\tilde{X}, \leq)$.

3.4.5 Control Module Generation

The purpose of extending the system description to $\tilde{\Sigma}$ and defining the partial order $(\tilde{X}, \leq)$ is to invoke previous work concerning two-vehicle cooperative safety control, see [36], and two-vehicle competitive safety control, see [49]. We look to now generate the modules defined in Section 3.3.4 with the extended system description.

Before invoking our previous results, we must first discuss the topological properties of the global bad set $\tilde{\mathbf{B}}$ and the local bad sets $\tilde{B}^k_M$ and $\tilde{B}^k_{RE}$, where $k \in \mathscr{I}$.

Order Preserving Connected Sets

We define a class of sets called order preserving connected (o.p.c.) in $\mathbb{R}^2$ with respect to the standard partial order $(\mathbb{R}^2, \leq)$. As defined in [49], order preserving connected sets have the property that any non-empty intersection with a pointed cone is a path connected set. An example of such a set is provided in Figure 3.8. Order preserving connected sets are useful in safety control problems defined with order preserving systems, in that they provide a natural geometry for capturing the flow. That is to say, once the forward cone of evolution is entirely pointed into the set, then the flow must necessary enter for any control input.

With some abuse of notation, for the two-vehicle index $k \in \mathscr{I}$ we will say the set

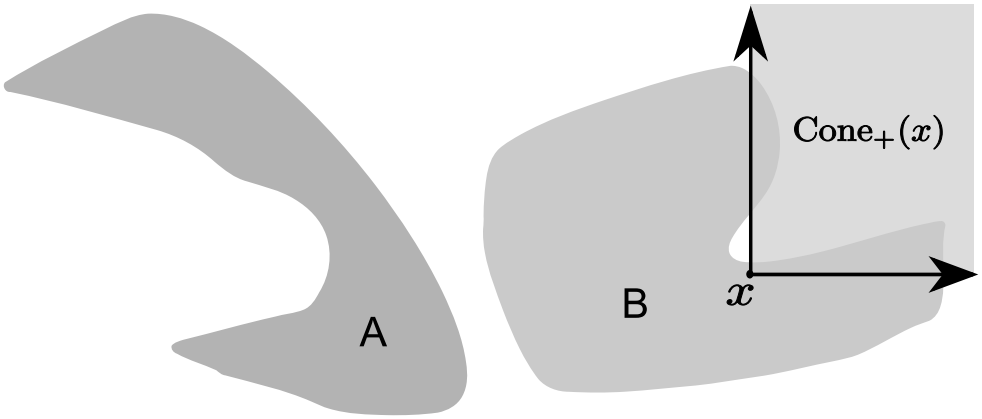


Figure 3.8: A is an o.p.c. set while B is not an o.p.c. set.

$A \subset \tilde{X}^k$ (which is a subset of $\mathbb{R}^4$) is o.p.c. provided the projection of A onto $\tilde{X}_1^k$ is o.p.c.

Connected Components of the Bad Set B

For vehicles $k \in \mathscr{I}$, the projection of the extended bad set $\tilde{B}_s^k$ into the extended space of constraint $\tilde{X}_1^k$ is not o.p.c. Therefore, in order to invoke prior results using o.p.c. sets, we instead look to solve the safety control problem for each connected component of $\tilde{B}_s^k$ when projected into the space of constraint $\tilde{X}_1^k$.

To isolate each connected component of $\tilde{X}_1^k$, we define the shifted box set. For two-vehicle index set $k \in \mathscr{I}$ and integers $l \in \mathbb{Z}^2$ we define the box set $Box^k(l) \subset \tilde{X}^k$ as

$$Box^k(l) := \{\tilde{x}^k \in \tilde{X}^k \mid x_1^k \in [l_1 D^{k_1}, (l_1+1)D^{k_1}] \times [l_2 D^{k_2}, (l_2+1)D^{k_2}].$$

We now use the box set to construct isolated components of the two-vehicle collision set $\tilde{B}_s^k$, where $s \in \{M, RE\}$ and $k \in \mathscr{I}$. Define the set

$$\tilde{B}_s^k(l) := \tilde{B}_s^k \cap Box^k(l)$$

which by the construction of $\tilde{B}_s^k$ is an o.p.c. set. Note that the union of all such sets gives us the entire bad set $\tilde{B}_s^k$, that is

$$\tilde{B}_s^k = \bigcup_{l \in \mathbb{Z}^2} \tilde{B}_s^k(l).$$

Therefore, if we can construct a module for $\tilde{B}_s^k(l)$, and show that the conjunction of each module is non-conflicting, then we can guarantee the entire safety specification $\tilde{B}_s^k$ is never violated.

3.4.6 Cooperative Module Solution

We seek to design a controller that guarantees the state of the system never enters $\tilde{\mathbf{B}}_s^k(l)$ for $s \in \{M, RE\}$ and $k \in \mathscr{I}$. We accomplish this by first looking for the primitive $g_s^k(\cdot,\cdot)_{coop} : \tilde{X}^k \times \mathbb{Z}^2 \rightrightarrows U^k$. We assume each vehicle can apply control to avoid a merging collision, therefore we look to design these primitives and extend them as cooperative modules, as defined in Section 3.3.4.

In order to find such a $g_s^k(\cdot,\cdot)_{coop}$ where $k \in \mathscr{I}$, we invoke previous results involving cooperative two vehicle collision avoidance [36]. We first look to compute the extended capture set

$$\tilde{C}_s^k \;:=\; \{x^k \in \tilde{X}^k \mid \forall\, \mathbf{u}^k \in S(U^k),\ \exists\, t \in \mathbb{R}_+ \text{ s.t } \tilde{\phi}^k(t, x^k, \mathbf{u}^k) \in \tilde{B}_s^k\}.$$

Rather than directly computing this capture set, we instead compute simpler sets. For a fixed control signal $\mathbf{u}^k \in S(U^k)$, we define the *restricted* capture set

$$\tilde{C}_s^k(\mathbf{u}^k) \;:=\; \{\tilde{x}^k \in \tilde{X}^k \mid \exists\, t \in \mathbb{R}_+ \text{ s.t. } \tilde{\phi}^k(t, \tilde{x}^k, \mathbf{u}^k) \in \tilde{B}_s^k\}.$$

The set $\tilde{C}_s^k(\mathbf{u}^k)$ corresponds to the sets of all states that are taken into $\tilde{B}_s^k$ under the fixed input $\mathbf{u}^k \in S(U^k)$.

With the partial order $(\tilde{X}, \leq)$ and order preserving property of the system given by (*ii*) from Definition 19, we can compute each capture set using the restricted capture sets if the bad set is o.p.c. In this case, the bad set $\tilde{B}_s^k$ is not o.p.c., however for the shift parameter $l \in \mathbb{Z}^2$ we know that $\tilde{B}_s^k(l)$ is o.p.c., as discussed in section 3.4.5.

For vehicles $k \in \mathscr{I}$, the safety specification $s \in \{M, RE\}$ and the shift parameter $l \in \mathbb{Z}^2$, we define the shifted capture set as

$$\tilde{C}_s^k(l) := \{x^k \in \tilde{X}^k \mid \forall \mathbf{u}^k \in S(U^k),\ \exists\, t \in \mathbb{R}_+ \text{ s.t } \tilde{\phi}^k(t, x^k, \mathbf{u}^k) \in \tilde{B}_s^k(l)\},$$

and for the input $\mathbf{u}^k \in S(U^k)$, we define the shifted restricted capture set as

$$\tilde{C}_s^k(\mathbf{u}^k, l) \quad := \quad \{\tilde{x}^k \in \tilde{X}^k \mid \exists\, t \in \mathbb{R}_+ \text{ s.t. } \tilde{\phi}^k(t, \tilde{x}^k, \mathbf{u}^k) \in \tilde{B}_s^k(l)\}.$$

For $i \in \{1, 2, 3\}$, we define the constant inputs $\mathbf{u}_L^i := u_L^i$ for all $t \in \mathbb{R}_+$ and $\mathbf{u}_H^i := u_H^i$ for all $t \in \mathbb{R}_+$. These signals can be used to construct the two useful input signals $\mathbf{u}_{\mathcal{H}} := (\mathbf{u}_L^1, \mathbf{u}_H^2, \mathbf{u}_H^3)$ and $\mathbf{u}_{\mathcal{L}} := (\mathbf{u}_H^1, \mathbf{u}_L^2, \mathbf{u}_L^3)$. With the shifted restricted capture set generated from a bad set that is o.p.c., we can now invoke a critical result using the signals $\mathbf{u}_{\mathcal{H}}$ and $\mathbf{u}_{\mathcal{L}}$.

Theorem 7. For the shift parameter $l \in \mathbb{Z}^2$, safety specification $s \in \{M, RE\}$ and index $k \in \mathscr{I}$, we have

$$\tilde{C}_s^k(l) \quad = \quad \tilde{C}_s^k(\mathbf{u}_{\mathcal{L}}^k, l) \cap \tilde{C}_s^k(\mathbf{u}_{\mathcal{H}}^k, l).$$

A proof of this Theorem can be found in [36]. This Theorem states if the initial condition $\tilde{x}^k$ generates a flow that enters the bad set $\tilde{B}_s^k(l)$ under both the inputs $\mathbf{u}_{\mathcal{H}}$ and $\mathbf{u}_{\mathcal{L}}$, then the flow will enter the bad set no matter what input is applied. From the assumptions on the system dynamics and the geometry of $\tilde{B}_s^k(l)$, the restricted capture set $\tilde{C}_s^k(\mathbf{u}^k, l)$ can be computed using linear complexity algorithms that are guaranteed to terminate. We provide this algorithm in Section 3.5.1.

From this characterization of the capture set, for $j \in \{2, 3\}$ we can construct the primitive in the form of the static feedback set-valued map $g_s^k(\cdot, l)_{coop} : \tilde{X}^k \rightrightarrows U^k$ to render the

set $\tilde{X}^k \backslash C_s^k(l)$ controlled invariant (see [36]), where

$$g_s^k(\tilde{x}^k, l)_{coop} := \begin{cases} u_{\mathcal{L}}^k & \text{if } \tilde{x}^k \in \partial \tilde{C}_s^k(\mathbf{u}_{\mathcal{L}}^k, l) \cap \tilde{C}_s^k(\mathbf{u}_{\mathcal{H}}^k, l), \\ u_{\mathcal{H}}^k & \text{if } \tilde{x}^k \in \partial \tilde{C}_s^k(\mathbf{u}_{\mathcal{H}}^k, l) \cap \tilde{C}_s^k(\mathbf{u}_{\mathcal{L}}^k, l), \\ \{u_{\mathcal{H}}^k, u_{\mathcal{L}}^k\} & \text{if } \tilde{x}^k \in \partial \tilde{C}_s^k(\mathbf{u}_{\mathcal{L}}^k, l) \cap \partial \tilde{C}_s^k(\mathbf{u}_{\mathcal{H}}^k, l), \\ U^k & \text{else.} \end{cases} \tag{3.2}$$

Under this primitive, control action is only applied when the state is inside one of the restricted capture set while simultaneously on the boundary of the other restricted capture set. If the state is in the intersection of the both restricted capture set boundaries, then two possible control actions are possible. Otherwise, any control input is acceptable.

We define the module $\bar{g}_s^k(\cdot, l, \cdot)_{coop} : \tilde{X} \times 2^X \rightrightarrows U$ for $j \in \{2, 3\}$ by

$$\bar{g}_s^k(\tilde{x}, l, \mathbf{A})_{coop} := \begin{cases} \{u \in U \mid u^k \in g_s^k(\tilde{x}, l)_{coop}\} & \text{if } \xi(\tilde{x}) \in \mathbf{A}, \\ U & \text{else,} \end{cases} \tag{3.3}$$

where $\tilde{x} \in \tilde{X}$ represents the extended state and $\mathbf{A} \subset X$ represents the enable set.

We next would like to show there exists a *termination* of the evaluation of $\bar{g}_s^k(\hat{y}, l, \mathbf{A})_{coop}$ when we set $\mathbf{A} = X$. Since this map itself is constructed from the restricted capture set $\tilde{C}_s^k(\mathbf{u}^k, l)$, we use these sets in the termination condition.

Termination of Cooperative Module

We next provide a measure of *termination* in the construction of $\bar{g}_M^k(\hat{y}, l, X)_{coop}$. Since we compute this module using the restricted capture sets $\tilde{C}_s^k(\mathbf{u}^k, l)$, we use them directly in this characterization.

We now use the geometric tool $\alpha_s^k(l) \in \tilde{X}_1^k$ to define the *termination* of the feedback set-valued map $\bar{g}_s^k(\hat{y}, l, X)_{coop}$ as

$$\alpha_s^k(l) := \inf\{\tilde{x}_1^k \in \tilde{X}_1^k \mid \exists\, y^k \in \partial \tilde{C}_s^k(l) \text{ s.t. } x_1^k = y_1^k\}.$$

We will give conditions in Section 3.5.1 for the existence of α_s^k for $s = M$. This condition can be used to define the termination of the construction of $\bar{g}_s^k(x, l, X)_{coop}$.

Remark 2. Given $l \in \mathbb{Z}^2$ and $\tilde{x} \in \tilde{X}$, if $\tilde{x}_1^k < \alpha_s^k(l)$, then it follows that

$$\bar{g}_s^k(x, l, X) = U$$

This result follows from the fact that if $\tilde{x}_1^k < \alpha_s^k(l)$, then necessarily $\bar{g}_s^k(x, l, X)_{coop} = U$ since $x \cap C_M^k = \emptyset$.

Non-Conflicting Condition on Cooperative Module

For the specification $s \in M, RE$ and two-vehicle index $k \in \mathscr{I}$, we know from the construction of $\tilde{B}_s^k$ that for $l \in \mathbb{Z}^2$ only one component $\tilde{B}_s^k(l)$ can be within the box set $Box^k(l)$. However, it is not necessarily the case that the restricted capture sets $\tilde{C}_s^k(\mathbf{u}_{\mathcal{H}}^k, l)$ and $\tilde{C}_s^k(\mathbf{u}_{\mathcal{L}}^k, l)$ are within the box set $Box^k(l)$. Since the restricted capture sets are used in the construction of $\bar{g}_s^k(x, l, X_1^k)_{coop}$, we would like to have a condition that guarantees that we never have conflicts between these modules when taking separate shifting parameters $l, \bar{l} \in \mathbb{Z}^2$.

That is, we would like to show that for all $l, \bar{l} \in \mathbb{Z}^2$, we always have that the conjunction

$$\bar{g}_s^k(\tilde{x}, l, X)_{coop} \cap \bar{g}_s^k(\tilde{x}, \bar{l}, X)_{coop} \neq \emptyset. \tag{3.4}$$

We next provide a result that guarantees this is the case. Let $\vec{0} \in \mathbb{Z}^2$ be the zero vector $\vec{0} = (0, 0)$.

Proposition 4. If $\sup \tilde{B}_s^k(\vec{0}) - \alpha_s^k(\vec{0}) > (D^1, D^j)$, then for all $l, \bar{l} \in \mathbb{Z}^2$ we must have the conjunction of two modules are non-conflicting, that is

$$\bar{g}_s^k(x, l, X) \cap \bar{g}_s^k(x, \bar{l}, X) \neq \emptyset.$$

3.4.7 Competitive Module Solution

We seek to design a controller that guarantees the information state never enters $\tilde{\mathbf{B}}_s^k(l)$ for $s \in \{M, RE\}$, $l \in \mathbb{Z}^2$ and $k \in \mathscr{I}$. We accomplish this by first looking for the primitive

$g_s^k(\cdot,\cdot)_\delta : \tilde{X}^k \times \mathbb{Z}^2 \rightrightarrows U^{k_2}$. Since we do not want this module to conflict with the merging collision module, we ask that vehicle k_2 prevent a rear end collision *for any* input u^{k_1} applied by vehicle k_1.

In order to find such a $g_{RE}^k(\cdot, l, X)_\delta$ for $k \in \mathscr{I}$ and $l \in \mathbb{Z}^2$, we invoke our previous results involving two-agent collision avoidance under the competitive control. That is we seek to find the capture set

$$\tilde{C}_s^k(l) \;:=\; \left\{\tilde{x}^k \in \tilde{X}^k \mid \forall\, \mathbf{u}^{k_2} \in \mathcal{U}^{k_2},\ \exists\, \mathbf{u}^{k_1} \in S(U^{k_1}) \text{ s.t. } \tilde{\phi}^k(\mathbb{R}_+, \tilde{x}^k, \mathbf{u}^k) \cap \tilde{B}_{RE}^k(l) \neq \emptyset\right\},$$

and the feedback set-valued map $g_{RE}^k(\cdot, l) : \tilde{X}^k \times \rightrightarrows U^j$ such that if the initial condition $\tilde{x}^k \in \tilde{X}^k$ starts outside of $W_{RE}^k(l) := \tilde{X}^k \backslash \tilde{C}_k^{RE}(l)$, then no matter what input is applied by vehicle 1, the flow never enters $\tilde{B}_{RE}^k(l)$.

For a fixed control signal $\mathbf{u}^{k_2} \in \mathcal{U}^{k_2}$, we denote the restricted capture set

$$\tilde{C}_s^k(\mathbf{u}^{k_2}, l) := \{\tilde{x}^k \in \tilde{X}^k \mid \exists\, \mathbf{u}^{k_1} \in S(U^{k_1}),\ \exists t \in \mathbb{R}_+ \text{ s.t. } \tilde{\phi}^k(t, \tilde{x}^k, \mathbf{u}^k) \in \tilde{B}_{RE}^k\}.$$

The set $\tilde{C}_s^k(\mathbf{u}^i, l)$ corresponds to the set of all states that are taken into $\tilde{B}_s^k(l)$ for some $\mathbf{u}^{k_1} \in \mathcal{U}^{k_1}$ under the fixed input $\mathbf{u}^{k_2}$. Under the order preserving hypothesis and the fact that $\tilde{B}_s^k(l)$ is order preserving connected (see Section 3.4.5), we can compute each capture set using the restricted sub-capture sets with arguments $\mathbf{u}_L^{k_2}$ and $\mathbf{u}_H^{k_2}$.

Theorem 8. For the specification $s \in \{M, RE\}$, shift parameter $l \in \mathbb{Z}^2$ and o.p.c. bad set $\tilde{B}_s^k(l)$, we have that

$$\tilde{C}_s^k(l) = \tilde{C}_s^k(\mathbf{u}_L^{k_2}, l) \cap \tilde{C}_s^k(\mathbf{u}_H^{k_2}, l).$$

A proof of this result can be found in [49]. This characterization again provides us with

the static feedback set-valued map $g_s^k(\cdot,\cdot)_\delta : \tilde{X}^k \rightrightarrows U^{k_1}$ given by

$$g_s^k(\tilde{x}^k, l)_\delta := \begin{cases} u_L^{k_2} & \text{if } \tilde{x}^k \in \tilde{C}_s^k(\mathbf{u}_H^{k_2}, l) \cap \partial\tilde{C}_s^k(\mathbf{u}_L^{k_2}, l), \\ u_H^{k_2} & \text{if } \tilde{x}^k \in \tilde{C}_s^k(\mathbf{u}_L^{k_2}, l) \cap \partial\tilde{C}_s^k(\mathbf{u}_H^{k_2}, l), \\ \{u_H^{k_2}, u_L^{k_2}\} & \text{if } \tilde{x}^k \in \partial\tilde{C}_s^k(\mathbf{u}_L^{k_2}, l) \cap \partial\tilde{C}_s^k(\mathbf{u}_H^{k_2}, l), \\ U^{k_2} & \text{else.} \end{cases} \tag{3.5}$$

We define the module $\bar{g}_s^k(\cdot, l, \cdot)_\delta : \tilde{X} \times X \rightrightarrows U$ with

$$\bar{g}_s^k(\tilde{x}, l, \mathbf{A})_\delta := \begin{cases} \{u \in U \mid u^{k_2} \in g_s^k(\tilde{x}^k, l)_\delta\} & \text{if } \xi(\tilde{x}) \in \mathbf{A}, \\ U & \text{else,} \end{cases} \tag{3.6}$$

where $\tilde{x} \in \tilde{X}$ represents the extended state and $\mathbf{A} \subset X$ represents the enable set.

3.4.8 Control Modules Used

We assume each vehicle can apply control to avoid a merging collision, therefore use cooperative modules to solve for the M specification types, as constructed in Section 3.4.6. We choose the enable sets $\mathbf{A}_M^k := X$ to be the entire state space.

We choose to satisfy the rear-end safety specification using the competitive modules. We choose the enable sets to be $\mathbf{A}_{RE}^k := [\inf B_{RE}^k, \sup B_{RE}^k]$, which represents the smallest rectangle set that completely inscribes the bad set B_{RE}^k.

3.4.9 Composition of the Control Modules

We combine control modules through conjunction of their extensions. In order to show that the conjunction maintains each modularized safety specification, we must show that the conjunction is non-blocking. We next provide a set of conditions that guarantees the conjunction of the previously mentioned modules is non-blocking.

To allow us to analyze the conjunction of the modules, we first ask for a condition on the maximum size of uncertainty in our system. If the set of uncertainty is allowed to grow

without bound (such is the case when communication channels fail), we cannot guarantee that the safety condition be maintained.

Conditions on Composition

We next provide a condition guaranteeing the system is able to avoid both merging collisions.

Lemma 2. *If* $U_M^{(1,2)} < \alpha_M^{(1,3)}$ *and* $U_M^{(1,3)} < \alpha_M^{(1,2)}$*, then* $\bar{g}_M(x) := \bar{g}_M^{(1,2)}(x) \cap \bar{g}_M^{(1,3)}(x)$ *is non-conflicting.*

Proof. We show that there cannot exist an $x \in X$ such that $\bar{g}_M^{(1,2)}(x) \cap \bar{g}_M^{(1,3)}(x) = \emptyset$.

We have that the input to agent 3 under the set-valued map $\bar{g}_M^{(1,2)}(x)$ always evaluates to U^3 for all $x \in X$, that is $\tau_3(\bar{g}_M^{(1,2)}(x)) = U^3$ for all $x \in X$. Thus, $\bar{g}_M^{(1,2)}(x)$ does not affect the input of agent 3. The same statement can be made regarding $\bar{g}_M^{(1,3)}$, namely that $\tau_2(\bar{g}_M^{(1,3)}(x)) = U^2$. Therefore, blocking can only occur with respect to the input of agent 1. Therefore, we will show that for all $x \in X$,

$$\tau_1(\bar{g}_M^{(1,2)}(x)) \cap \tau_1(\bar{g}_M^{(1,3)}(x)) \neq \emptyset. \tag{3.7}$$

Suppose that $\tau_1(\bar{g}_M^{(1,2)}(x)) \neq U^1$. We know that if $\phi_1^{1,2}x(t, x_0, \mathbf{u})) < \alpha_{12}$, then we must necessarily have that $\bar{g}_M^{(1,2)}(x) = U$ by the definition of $\alpha_M^{(1,2)}$. Similarly, the order preserving condition of the dynamics with respect to initial condition implies that if $U_M^{(1,2)} < \phi_1^{(1,2)}(t, x_0, \mathbf{u})$, then we must also have $\bar{g}_M^{(1,2)}(x) = U$. Therefore, we must have that

$$\phi_1^{(1,2)}(t, x_0, \mathbf{u}) \in [\alpha_M^{(1,2)}, U_M^{(1,2)}]. \tag{3.8}$$

If (3.8) holds, the assumption $U_M^{(1,2)} < \alpha_M^{(1,3)}$ in the Lemma statement implies that

$$\phi_1^1(t, x_0, \mathbf{u}) < \alpha_M^{(1,3)}. \tag{3.9}$$

If (3.8) holds, the assumption $U_M^{(1,3)} < \alpha_M^{(1,2)}$ in the Lemma statement that

$$U_M^{(1,3)} < \phi_1^{1,2}(t, x_0, \mathbf{u}, \mathbf{z}). \tag{3.10}$$

To accommodate the situation where $\alpha_M^{(1,2)}$ does not wrap around, the previous statement would read $U_M^{(1,3)} < (D^1, D^2) + \phi_1^{1,2}(t, x_0, \mathbf{u})$.

We can then combine (3.10)-(3.9), taking into account wrapping in $X_1^{(1,2)}$, to get $\phi(1,2)_1(t, x_0, \mathbf{u}) \cap [\alpha^{(1,3)}, U_M^{(1,3)}] = \emptyset$. This along with the definition of $\alpha_M^{(1,2)}$ and the order preserving property of the dynamics with respect to initial condition implies that $\bar{g}_M^{(1,3)}(x) = U$.

Therefore, if $\bar{g}_M^{(1,2)}(x) \neq U$ then we must have $\bar{g}_M^{(1,3)}(x) = U$, implying that (3.7) must always hold, implying that $\bar{g}^M = \bar{g}_M^{(1,2)}(x) \cap \bar{g}_M^{(1,3)}(x) \neq \emptyset$ for all $x \in X$. □

For $k \in \mathscr{I}$, define the sets $\overline{D}^k := \{x^k \in X^k \mid x_1^k = (L_M^k, U_M^k)\}$ and $\underline{D}^k := \{x^k \in X^k \mid x_1^k = (U_M^k, L_M^k)\}$. We next provide a condition guaranteeing that the composition of modules g_M^k and g_{RE}^k are non-conflicting.

Lemma 3. If $\overline{D}^k \cap C_{RE}^k = \emptyset$ and $\underline{D}^k \cap C_{RE}^k = \emptyset$, then $\bar{g}^k(x) := \bar{g}_{RE}^k(x) \cap \bar{g}_M^k(x) \neq \emptyset$ for all $x \in X$.

Proof. We carry out the proof for $\bar{g}_{12}(x)$, a similar proof holds for $\bar{g}_{13}(x)$. Since $g_{RE}^{1,2}$ only maps into U^2 and $g_M^{(1,2)}$ only maps into $U^{1,2}$, the only possible conflict between $\bar{g}_{RE}^{1,2}(x)$ and $\bar{g}_M^{(1,2)}(x)$ is into the input U^2. Therefore, we will show that for all $x \in X$,

$$\tau_2(\bar{g}_M^{(1,2)}(x)) \cap \tau_2(\bar{g}_{RE}^{(1,2)}(x)) \neq \emptyset. \tag{3.11}$$

From the definition g_{RE}^k, we know that the evaluation becomes non-trivial (equal to U^2) only after the flow $\phi(t, x, \mathbf{u})$ enters the set $Av_{RE}^{1,2}$. Conversely, we know that once the flow enters the set $Av_{RE}^{1,2}$, the order preserving property on the dynamics implies that unless the

flow is inside the bad set $B_M^{(1,2)}$, it is impossible to have a merging collision until the agents once again approach the intersection.

If the module $\bar{g}_M^{(1,2)}(x)$ properly enforces the merging collision safety specification, then order preserving property with respect to the dynamics implies the flow must pass either above or below the bad set $B_M^{(1,2)}$, when projected on the space of constraint. □

We now combine each of these results to show that the complete system is safe.

Theorem 9. *If $\bar{g}_{RE}(x)$ and $\bar{g}^M(x)$ are each defined for all $x \in X$, then $\bar{g}(x) := \bar{g}_{RE}^{1,2}(x) \cap \bar{g}_M^{(1,2)}(x) \cap \bar{g}_{RE}^{(1,3)}(x) \cap \bar{g}_M^{(1,3)}(x)$ is non-conflicting, that is $\bar{g}(x) \neq \emptyset$ for all $x \in X$.*

Proof. We first combine the modules $\tilde{\bar{g}}(x) := \bar{g}_{RE}^{1,2}(x) \cap \bar{g}_M^{(1,2)}(x) \cap \bar{g}_{RE}^{(1,3)}(x)$. This conjunction is non-conflicting since $\bar{g}_{RE}^{1,2}(x) \cap \bar{g}_M^{(1,2)}(x) = \bar{g}_{12}(x)$ only constrains $U^{1,2}$ while $\bar{g}_{RE}^{(1,3)}(x)$ only constrains U^3. If we add a module to $\tilde{\bar{g}}(x)$, we need only check that the conjunction is non-conflicting by checking: (i) Image into $U^{1,2}$ does not conflict with $\bar{g}_{12}(x)$, (ii) Image into U^3 does not conflict with $\bar{g}_{RE}^{(1,3)}(x)$.

We add the module $\bar{g}_M^{(1,3)}(x)$ to $\tilde{\bar{g}}(x)$ through conjunction and show that $\bar{g}_{RE}^{(1,3)}(x)$ does not conflict with any of the modules that make up $\tilde{\bar{g}}(x)$. This will imply the entire conjunction is non-blocking.

(i) We have already shown that $\bar{g}_M^{(1,3)}(x) \cap \bar{g}_M^{(1,2)}(x)$ is non-conflicting by the definition of $\bar{g}^M(x)$. Furthermore, we have that if $\bar{g}_M^{(1,2)}(x) \neq U$, then necessarily $\bar{g}_M^{(1,3)}(x) = U$. Thus any such $x \in X$ that generates a non-trivial evaluation $\bar{g}_{12}(x) \neq U$, we must have that $\bar{g}_M^{(1,3)}(x) = U$. This implies that $\bar{g}_M^{(1,3)}(x) \cap \bar{g}_{RE}^{1,2}(x)$ is non-conflicting.

(ii) We have already shown that $\bar{g}_M^{(1,3)}(x) \cap \bar{g}_{RE}^{(1,3)}(x)$ is non-conflicting by the definition of $\bar{g}_{13}(x)$.

Since the above conditions (i) and (ii) both hold for all $x \in X$, we must have that the entire composition is non-conflicting, that is $\bar{g}(x) \neq \emptyset$ for all $x \in X$. □

3.5 Algorithms

We first compute the discrete time equivalent of the dynamics defined in Section 3.2, since they will be used in the computation of restricted capture sets $C_s^k(\mathbf{u})$. For the i^{th} agent with $i \in \{1, 2, 3\}$, the discrete time equations (employing forward Euler approximation) are given with step size $\Delta T > 0$ and index $m \in \mathbb{N}$ as

$$\begin{aligned} x_1^i[m+1] &= x_1^i[m] + F^i(x_2^i[m], \mathbf{u}^i[m]), \\ x_2^i[m+1] &= \bar{F}^i(x_2^i[m], \mathbf{u}^i[m]), \end{aligned}$$

where $F^i(x_2^i[m], \mathbf{u}^i[m]) := \Delta T x_2^i[m]$ and $\bar{F}^i(x_2^i[m], \mathbf{u}^i[m]) := x_2^i[m] + \Delta T(\alpha^i)$ with α^i as defined in the hybrid automaton seen in Figure 3.2.

For the input signal $\mathbf{u}^i \in \mathcal{U}^i$ to agent $i \in \{1, 2, 3\}$, define the recursive sequence $\bar{F}^i : X_2^i \times S(\mathcal{U}^i) \times \mathbb{N} \to X_1^i$ as

$$\begin{aligned} \bar{F}^i(x_2^i, \mathbf{u}^i[-1])_0 &:= x_2^i, \\ \bar{F}^i(x_2^i, \mathbf{u}^i[m])_{m+1} &:= \bar{F}^i(\bar{F}^i(x_2^i, \mathbf{u}^i[m-1])_m, \mathbf{u}^i[m]). \end{aligned}$$

We can compute the displacement $x_1^i[m]$ for each vehicle $i \in \{1, 2, 3\}$ from the initial condition $x^i \in X^i$ as

$$x_1^i[m] = x_1^i + \sum_{j=0}^{m-1} F^i(\bar{F}^i(x_2^i, \mathbf{u}^i[j-1])_j, \mathbf{u}^i[j]).$$

We introduce the sequence ζ^i for vehicle $i \in \{1, 2, 3\}$ for a velocity $x_2^i \in X_2^i$ and constant input signal $\mathbf{u}^i \in S(\mathcal{U}^i)$ as

$$\zeta^i(x_2^i, \mathbf{u}^i)_m := \sum_{j=0}^{m-1} F^i(\bar{F}^i(x_2^i, u^i)_j, u^i),$$

giving $x_1^i[m] = x_1^i + \zeta^i(x_2^i, \mathbf{u}^i)_m$.

3.5.1 Computation of Restricted Capture Sets for CA Collisions

For $k \in \mathscr{I}$ with $i \in \{k_1, k_2\}$, the constant input u^i and the initial condition x_2^i, let

$$\begin{aligned} L_M^i(x_2^i, u^i)_m &:= L_M^k - \zeta^i(x_2^i, u^i)_m, \\ U_M^i(x_2^i, u^i)_m &:= U_M^k - \zeta^i(x_2^i, u^i)_m. \end{aligned}$$

We note that these sequences are order reversing in their arguments.

Theorem 10.

$$C_M^k(\mathbf{u}) = \left\{ x^k \in X^k \mid \exists\, m \in \mathbb{Z}_+ \text{ s.t. } L_M^k(x_2^k, u^k)_m < x_1^k < U_M^k(x_2^k, u^k)_m \right\}.$$

A proof of this result can be found in [36]. It has been shown that these algorithms are *guaranteed* to terminate in a finite number of iterations $m \in \mathbb{Z}_+$.

Computation of Bound on CA Restricted Capture Set

We have previously defined the bound α^k in Section 3.4.9, we now give an existence condition and a means of computation.

Lemma 4. If (i) $a^i u_H^i + b^i > 0$; (ii) $a^i u_L^i + b^i < 0$ hold for $i \in \{1, 2, 3\}$, and $v_{min}^i < v_{max}^j$ for all $i, j \in \{1, 2, 3\}$, then α_k for all $k \in \mathscr{I}$ exist and are finite.

Proof. We show the result with respect to $\alpha^{(1,2)}$, a similar proof can be carried out for $\alpha^{(1,3)}$. Theorem 7 allows us to determine $C_M^{(1,2)}$ with $C_M^{(1,2)}(\mathbf{u}_{\mathcal{L}}^{(1,2)})$ and $C_M^{(1,2)}(\mathbf{u}_{\mathcal{H}}^{(1,2)})$.

While computing $C_M^{(1,2)}(\mathbf{u}_{\mathcal{L}}^{(1,2)})$, assumption (i) of the Lemma hypothesis implies that $\zeta^1(x_2^1, u_H^1)_m$ is a strictly monotone sequence of partial sums which achieves the upper bound v_{max}^1 in finite m. Similarly, assumption (ii) of the Lemma hypothesis implies that $\zeta^2(x_2^2, u_L^2)_m$ reaches the lower bound v_{min}^2 in finite m. Likewise, $\zeta^1(x_2^1, u_L^1)_m$ tends to v_{min}^1 and $\zeta^2(x_2^2, u_H^2)_m$ tends to v_{max}^2 in finite steps.

Since $v^1_{min} < v^1_{max}$, we must have a step m_1 such that (a) $U^1_M(x^1_2, u^1_H)_{m_1} < L^1_M(x^1_2, u^1_L)_{m_1}$ for any velocity x^1_2. Since $v^2_{min} < v^2_{max}$, we must have a step m_2 such that (b) $U^2_M(x^2_2, u^2_H)_{m_2} < L^2_M(x^2_2, u^2_L)m_2$ for any velocity x^2_2. Once achieved, these inequalities will be maintained by the order reversing properties, that is $U^1_M(x^1_2, u^1_H)_m < L^1_M(x^1_2, u^1_L)_m$ for $m \geq m_1$ and $U^2_M(x^2_2, u^2_H)_m < L^2_M(x^2_2, u^2_L)_m$ for $m \geq m_2$.

For a given velocity x_2, consider the rectangles used in the computation of the slices of $C^{(1,2)}_M(\mathbf{u}^{1,2}_{\mathcal{L}})$ and $C^{(1,2)}_M(\mathbf{u}^{1,2}_{\mathcal{H}})$. For notation, define these rectangles respectively as

$$R(x_2, \mathbf{u}_{\mathcal{L}})_m := [L^1_M(x^1_2, u^1_H)_m, U^1_M(x^1_2, u^1_H)_m] \times [L^2_M(x^2_2, u^2_L)_m, U^2_M(x^2_2, u^2_L)_m], \tag{3.12}$$

$$R(x_2, \mathbf{u}_{\mathcal{H}})_m := [L^1_M(x^1_2, u^1_L)_m, U^1_M(x^1_2, u^1_L)_m] \times [L^2_M(x^2_2, u^2_H)_m, U^2_M(x^2_2, u^2_H)_m]. \tag{3.13}$$

Now we take $\bar{m} := \max(m_1, m_2)$. It follows from the inequalities (a) and (b) that $R(x_2, \mathbf{u}_{\mathcal{L}})_{\bar{m}} \cap R(x_2, \mathbf{u}_{\mathcal{H}})_{\bar{m}} = \emptyset$. Now we show the intersection is empty over the union of all subsequent iterations, that is

$$\bigcup_{m \geq \bar{m}} R(x_2, \mathbf{u}_{\mathcal{L}})_m \cap \bigcup_{m \geq \bar{m}} R(x_2, \mathbf{u}_{\mathcal{H}})_m = \emptyset. \tag{3.14}$$

Suppose that this does not hold, then there are $\bar{m}_1, \bar{m}_2$ greater than $\bar{m}$ such that $R(x_2, \mathbf{u}_{\mathcal{L}})_{\bar{m}_1} \cap R(x_2, \mathbf{u}_{\mathcal{H}})_{\bar{m}_2} \neq \emptyset$. This implies that both (c) $L^1_M(x^1_2, u^1_L)_{\bar{m}_2} < U^1_M(x^1_2, u^1_H)_{\bar{m}_1}$ and (d) $L^2_M(x^2_2, u^2_L)_{\bar{m}_1} < U^2_M(x^2_2, u^2_H)_{\bar{m}_2}$ hold.

If $\bar{m}_1 \leq \bar{m}_2$, then $U^2_M(x^2_2, u^2_H)_{\bar{m}_2} \leq U^2_M(x^2_2, u^2_H)_{\bar{m}_1}$ by the order reversing property. We know that (b) implies that $U^2_M(x^2_2, u^2_H)_{\bar{m}_1} < L^2_M(x^2_2, u^2_L)_{\bar{m}_1}$, therefore (d) cannot hold.

If instead $\bar{m}_2 \leq \bar{m}_1$, then $U^1_M(x^1_2, u^1_H)_{\bar{m}_1} \leq U^1_M(x^1_2, u^1_H)_{\bar{m}_2}$ by the order reversing property. We know that (a) implies that $U^1_M(x^1_2, u^1_H)_{\bar{m}_2} < L^1_M(x^1_2, u^1_L)_{\bar{m}_2}$, therefore (c) cannot hold.

Therefore, (c) and (d) cannot both be true, implying (3.14) always holds given an arbitrary velocity x_2. Thus, the intersection of $C^{(1,2)}_M(\mathbf{u}^{(1,2)}_{\mathcal{L}})$ and $C^{(1,2)}_M(\mathbf{u}^{(1,2)}_{\mathcal{H}})$ terminates for each velocity x_2 with a non-zero velocity, where $L^1_M(x^1_2, u^1_L)_{\bar{m}} \leq \alpha^{(1,2)}$. Since x_2 is an element of

the compact set X_2, we must have that $L_M^1(x_2^1, u_L^1)_{\bar{m}}$ is defined for all x_2, thus $\alpha^{(1,2)}$ exists and is bounded. □

3.5.2 Computation of Restricted Capture Sets for Rear-End Collisions

Unlike B_M^k which is an interval set within the space of constraint, B_{RE}^k is a more general o.p.c. sets within the space of constraint.

Computation of Restricted Capture Set $\tilde{C}_{RE}^k(\mathbf{u}^j, \vec{0})$

We note that B_{RE}^k is completely characterized by the positive constants L_{RE}^k, U_{RE}^k and l. For the constant input u^{k_1} and initial condition x_2^k, we define the sequences with index $m \in \mathbb{N}$

$$\begin{aligned}
l(x_2^{1,k_2}, u^{k_2})_m &:= -l - \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m, \\
h(x_2, u^{k_2})_m &:= l - \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m, \\
L_{RE}^{k_1}(x_2^{k_1})_m &:= L_{RE}^k - \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m, \\
U_{RE}^{k_1}(x_2^{k_1})_m &:= U_{RE}^k - \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m, \\
L_{RE}^{k_2}(x_2^{k_2}, u^{k_2})_m &:= L_{RE}^k - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m, \\
U_{RE}^{k_2}(x_2^{k_2}, u^{k_2})_m &:= U_{RE}^k - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m.
\end{aligned}$$

We can compute the restricted capture set $C_{RE}^k(u^{k_2})_m$ with the above sequences.

Claim 2.

$$C_{RE}^k(u^{k_2})_m = \left\{ x^k \in X^k \;\middle|\; \begin{array}{l} \exists\, m \in \mathbb{Z}_+ \text{ s.t. } (x_1^{k_1} - x_1^{k_2}) \in (l^{k_2}(x_2^k, u^{k_2})_m, h^{k_2}(x_2, u^{k_2})_m), \\ L_{RE}^{k_2}(x_2^{k_1}, u^{k_2})_m < x_1^{k_2} < U_{RE}^{k_2}(x_2^{k_1}, u^{k_2})_m \\ \text{and } L_{RE}^{k_1}(x_2^{k_1})_m < x_1^{k_1} < U_{RE}^{k_1}(x_2^{k_1})_m \end{array} \right\}.$$

Proof. Denote

$$S := \left\{ x^k \in X^k \;\middle|\; \begin{array}{l} \exists\, k \in \mathbb{Z}_+ \text{ s.t. } (x_1^{k_1} - x_1^{k_2}) \in (l^{k_2}(x_2^k, u^{k_2})_m, h^{k_2}(x_2, u^{k_2})_m), \\ L_{RE}^{k_2}(x_2^{k_2}, u^{k_2})_m < x_1^{k_2} < U_{RE}^{k_2}(x_2^{k_2}, u^{k_2})_m \text{ and } L_{RE}^{k_1}(x_2^{k_1})_m < x_1^{k_1} < U_{RE}^{k_1}(x_2^{k_1})_m \end{array} \right\},$$

we will show that $C^k_{RE}(u^{k_2})_m \subseteq S$ and $C^k_{RE}(u^{k_2})_m \supseteq S$.

($\subseteq$) Consider an element $x \in C^k_{RE}(u^{k_2})_m$, by definition there must exist an index $k \in \mathbb{Z}_+$ and input $\mathbf{u}^{k_1} \in S(U^{k_1})$ such that

$$x^k_1 + (\zeta^{k_1}(x^{k_1}_2, \mathbf{u}^{k_1})_m, \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m) \in \tilde{B}^k_{RE}. \tag{3.15}$$

By the order preserving properties of the dynamics with respect to input, we must have that $x^{k_1}_1 + \zeta^{k_1}(x^{k_1}_2, \mathbf{u}^{k_1})_m \in x^{k_1}_1 + [\zeta^{k_1}(x^{k_1}_2, u^{k_1}_L)_m, \zeta^{k_1}(x^{k_1}_2, u^{k_1}_H)_m]$. This statement can be combined with (3.15) to give

$$x^k_1 + [\zeta^{k_1}(x^{k_1}_2, u^{k_1}_L)_m, \zeta^{k_1}(x^{k_1}_2, u^{k_1}_H)_m] \times \{\zeta^{k_2}(x^{k_2}_2, u^{k_2})_m\} \cap \tilde{B}^k_{RE} \neq \emptyset.$$

The non-empty intersection with $\tilde{B}^k_{RE}$ holds if and only if

$$\begin{aligned}
&x^{k_1}_1 + [\zeta^{k_1}(x^{k_1}_2, u^{k_1}_L)_m, \zeta^{k_1}(x^{k_1}_2, u^{k_1}_H)_m] \cap (L^k_{RE}, U^k_{RE}) \neq \emptyset,\\
&x^{k_2}_1 + \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m \cap (L^k_{RE}, U^k_{RE}) \neq \emptyset,\\
&|x^{k_1}_1 + [\zeta^{k_1}(x^{k_1}_2, u^{k_1}_L)_m, \zeta^{k_1}(x^{k_1}_2, u^{k_1}_H)_m] - (x^{k_2}_1 + \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m)| \cap [0, l) \neq \emptyset.
\end{aligned}$$

The last statement can be reorganized to

$$(x^{k_1}_1 - x^{k_2}_1) + [\zeta^{k_1}(x^{k_1}_2, u^{k_1}_L)_m - \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m, \zeta^{k_1}(x^{k_1}_2, u^{k_1}_H)_m - \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m] \cap (-l, l) \neq \emptyset \tag{3.16}$$

For the simple case where $(x + [a, b]) \cap (c, d) \neq \emptyset$, it is clear that this holds if and only if $x > c - b$ and $x < d - a$. Therefore, statement (3.16) can be reorganized to read

$$\begin{aligned}
(x^{k_1}_1 - x^{k_2}_1) &> -l - \zeta^{k_1}(x^{k_1}_2, u^{k_1}_H)_m + \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m = l^{k_2}(x^k_2, u^{k_2})_m,\\
(x^{k_1}_1 - x^{k_2}_1) &< l - \zeta^{k_1}(x^{k_1}_2, u^{k_1}_L)_m + \zeta^{k_2}(x^{k_2}_2, u^{k_2})_m = h^{k_2}(x^k_2, u^{k_2})_m,
\end{aligned}$$

which combined give $(x^{k_1}_1 - x^{k_2}_1) \in (l^{k_2}(x^k_2, u^{k_2})_m, h^{k_2}(x_2, u^{k_2})_m)$, therefore x^k_1 is in S.

($\supseteq$) Let $x \in S$, by definition of S there must exist an index $k \in \mathbb{Z}_+$ such that the following inequalities are satisfied

$$L^k_{RE} - \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m < x_1^{k_1} < U^k_{RE} - \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m, \tag{3.17}$$

$$L^k_{RE} - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m < x_1^{k_2} < U^k_{RE} - \zeta^{k_2}(x_2^{k_1}, u^{k_2})_m, \tag{3.18}$$

$$(x_1^{k_1} - x_1^{k_2}) \in (-l - \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m, l - \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m) \tag{3.19}$$

From (3.18), we immediately have that

$$L^k_{RE} < x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m < U^k_{RE}, \tag{3.20}$$

therefore to show $x \in C^k_{RE}(u^{k_2})_m$ we would like to show the existence of an input $\mathbf{u}^{k_1} \in S(U^{k_1})$ such that

$$|x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m - x_1^{k_2} - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m| < l,\ L^k_{RE} < x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m < U^k_{RE}.$$

These inequalities can be combined to give the two inequalities

$$\zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m < \min\{U^k_{RE} - x_1^{k_1}, l - x_1^{k_1} + x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m\}, \tag{3.21}$$

$$\zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m > \max\{L^k_{RE} - x_1^{k_1}, -l - x_1^{k_1} + x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m\}. \tag{3.22}$$

For (3.19), we can rearrange terms

$$\begin{aligned} & (x_1^{k_1} - x_1^{k_2} - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m) \in (-l - \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m, l - \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m), \\ \Rightarrow\ & -l < x_1^{k_1} - x_1^{k_2} - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m + \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m \\ & \text{and } x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m - x_1^{k_2} - \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m < l \end{aligned}$$

The last two inequalities along with (3.17) give the inequalities

$$\begin{aligned} \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m &< \min\{U^k_{RE} - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} + l\}, \\ \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m &> \max\{L^k_{RE} - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} - l\}. \end{aligned}$$

Now suppose that either

$$\begin{aligned} \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m &\leq \min\{U_{RE}^k - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} + l\}, \\ \text{or} \quad \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m &\geq \max\{L_{RE}^k - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} - l\}, \end{aligned}$$

then we immediately satisfy (3.21)-(3.22) implying either $(x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m, x_2^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m) \in B_{RE}^k$ or $(x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m) \in B_{RE}^k$. Therefore, we must have that

$$\begin{aligned} \zeta^{k_1}(x_2^{k_1}, u_H^{k_1})_m &> \min\{U_{RE}^k - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} + l\}, \\ \text{and} \quad \zeta^{k_1}(x_2^{k_1}, u_L^{k_1})_m &< \max\{L_{RE}^k - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} - l\}. \end{aligned}$$

We have that $\zeta^{k_1}(x_2^{k_1}, \cdot) : S(U^{k_1}) \to X_1^{k_1}$ is a continuous function and $S(U^{k_1})$ is a connected subset of a metric space. Therefore, the intermediate value theorem implies we can find an input $\mathbf{u}^{k_1} \in S(U^{k_1})$ such that

$$\begin{aligned} \max\{L_{RE}^k - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} - l\} &< \zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m \\ &< \min\{U_{RE}^k - x_1^{k_1}, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m - x_1^{k_1} + l\}, \end{aligned}$$

giving

$$\max\{L_{RE}^k, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m\} < x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m < \min\{U_{RE}^k, x_1^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m + l\}.$$

The above inequality implies that (3.21)-(3.22) hold, which along with (3.20) implies that $(x_1^{k_1} + \zeta^{k_1}(x_2^{k_1}, \mathbf{u}^{k_1})_m, x_2^{k_2} + \zeta^{k_2}(x_2^{k_2}, u^{k_2})_m) \in B_{RE}^k$. Therefore, we have that $x \in C_{RE}^k(u^{k_2})$. □

3.5.3 State Estimation

Since the dynamics are order preserving with respect to the state, we can construct a state estimator that only keeps track of lower and upper bounds of the information state $\hat{x}(n, \hat{x}_0, \mathbf{u}[n], \mathbf{z}[n])$ similar to the estimator proposed in [37]. For notation, we let

$\vee\hat{x} := \sup \hat{x}$ and $\wedge\hat{x} := \inf \hat{x}$, which is taken component-wise. Consider the received message denotes the received message for the i^{th} vehicle, and with abuse of notation, let $h(\mathbf{z}^i[n]) := h(\mathbf{z}^i[n])$ where $\mathbf{z}^i[n] = (\mathbf{z}^i[n], \tau^i)$. We let $F^i(x^i[n], \mathbf{u}^i[n]) := (x_1^i[n] + \Delta T x_2^i[n], \bar{F}^i(x_2^i[n], \mathbf{u}^i[n]))$ for $i \in \{1, 2, 3\}$. We design with the maximum communication delay tolerance $\mu > 0$, where we do not use the most recent received signal if the time delay is greater than μ.

Estimation of Next State with Sensor Information

We can now design a state estimator that updates $\vee\hat{x}[n+1]$ and $\wedge\hat{x}[n+1]$ as follows. Consider vehicle $i \in \{1, 2, 3\}$, we assume that at all times the measurement is current, that is $\tau^i[n] = n$.

Then for $\hat{x}^i[n+1]$, we have

$$\begin{aligned} \vee\hat{x}^i[n+1] &= \inf\{F^i(\vee\hat{x}^i[n], \mathbf{u}^i[n]), \sup h^i(\mathbf{z}^i[n+1])\}, \\ \wedge\hat{x}^i[n+1] &= \sup\{F^i(\wedge\hat{x}^i[n], \mathbf{u}^i[n]), \inf h^i(\mathbf{z}^i[n+1])\}. \end{aligned}$$

For $k \in \mathscr{I}$, we must check to see that the received measurement is recent enough to use, otherwise we cannot use any measurement information. That is, for the measurements $\mathbf{z}^{k_1}[n]$ and $\mathbf{z}^{k_2}[n]$, the corresponding delays $\tau^{k_1} \in \mathbf{z}^i[n]$ and $\tau^{k_2} \in \mathbf{z}^j[n]$, we can compute $\vee\hat{x}^k[n+1]$ and $\wedge\hat{x}^{k_2}[n+1]$ with the algorithm

if $|\tau^{k_1} - \tau^j| \leq \mu$ **then**

$\vee\hat{x}^{k_2}[n+1] = \inf\{F^{k_2}(\vee\hat{x}^{k_2}[n], \mathbf{u}^{k_2}[n]), \sup h^{k_2}(\mathbf{z}^{k_2}[n+1])\}$

$\wedge\hat{x}^{k_2}[n+1] = \sup\{F^{k_2}(\wedge\hat{x}^{k_2}[n], \mathbf{u}^{k_2}[n]), \inf h^{k_2}(\mathbf{z}^{k_2}[n+1])\}$

else

$\vee\hat{x}^j[n+1] = F^j(\vee\hat{x}^j[n], \mathbf{u}^j[n])$

$\wedge\hat{x}^j[n+1] = F^j(\wedge\hat{x}^j[n], \mathbf{u}^j[n])$

end if

In this case, a received signal that is considered expired implies that the measurement $\mathbf{z}^{k_2}[n]$ for vehicle k_2 is equivalent to no measurement, that is $h^{k_1}(\mathbf{z}^{k_2}[n]) = X^{k_2}$.

Estimation of Future States

Similar to expired received signals, when projecting the dynamics into the future we must assume no measurement information. That is for vehicle $i \in \{1, 2, 3\}$ at time step $n \in \mathbb{Z}_+$, for any future time step $l \in \mathbb{N}$, we have the estimate given by

$$\begin{aligned} \vee\hat{x}^i[n+l] &= F^i(\vee\hat{x}^i[n+l-1], \mathbf{u}^i[n+l-1]), \\ \wedge\hat{x}^i[n+l] &= F^i(\wedge\hat{x}^i[n+l-1], \mathbf{u}^i[n+l-1]). \end{aligned}$$

Intersection with Capture Set

At each time step $n \in \mathbb{Z}_+$, we only need to check whether $[\wedge\hat{x}[n], \vee\hat{x}[n]]$ intersects with restricted capture sets. For the restricted capture sets $C_M^k(u^k)$, the fact that $L_M^i(x_2^i, u^i)_m$ and $U_M^i(x_2^i, u^i)_m$ are order reversing in the argument x_1^i implies that a sufficient condition guaranteeing for $k \in \mathscr{I}$ that $[\wedge\hat{x}^k[n], \vee\hat{x}^k[n]] \cap C_M^k(u^k) = \emptyset$ is that

$$[\wedge\hat{x}[n], \vee\hat{x}[n]] \cap [L_M^k(\wedge\hat{x}_2^k[n], u^k)_m, U_M^k(\wedge\hat{x}_2^k[n], u^k)_m] = \emptyset, \quad \forall\, m \in \mathbb{Z}_+.$$

3.6 Experimental Results

We present our experimental results on an autonomous roundabout system with 3 vehicles in which the proposed algorithms are implemented on-board each vehicle. We show the system parameters of the experimental setup satisfy the necessary conditions guaranteeing global safety. This is validated through experimental data with a duration of 26 minutes.

3.6.1 Experimental Setup

We introduce the experimental roundabout system with a description of the vehicles, positioning system, and communication. We consider 3 different experimental bad set con-

Bad Set Parameters for Configuration 1			
$B_M^{(1,2)}$	$L_M^{(1,2)}$ = (8.96, 2.18) m	$U_M^{(1,2)}$ = (10.76, 3.98) m	n/a
$B_M^{(1,3)}$	$L_M^{(1,3)}$ = (1.97, 2.42) m	$U_M^{(1,3)}$ = (2.87, 3.32) m	n/a
$B_{RE}^{1,2}$	$L_{RE}^{(1,2)}$ = (8.96, 2.18) m	$U_{RE}^{(1,2)}$ = (0.87, 5.2) m	l = .40 m
$B_{RE}^{(1,3)}$	$L_{RE}^{(1,3)}$ = (1.97, 2.42) m	$U_{RE}^{(1,3)}$ = (8.72, 9.17) m	l = .40 m

Table 3.1: The parameters used to construct each Bad Set (Section 3.3.1) are provided for Experimental Configuration 1.

Bad Set Parameters for Configuration 2			
$B_M^{(1,2)}$	$L_M^{(1,2)}$ = (8.96, 2.18) m	$U_M^{(1,2)}$ = (10.76, 3.98) m	n/a
$B_M^{(1,3)}$	$L_M^{(1,3)}$ = (1.97, 2.42) m	$U_M^{(1,3)}$ = (2.87, 3.32) m	n/a
$B_{RE}^{1,2}$	$L_{RE}^{(1,2)}$ = (8.96, 2.18) m	$U_{RE}^{(1,2)}$ = (0.87, 5.2) m	l = .40 m
$B_{RE}^{(1,3)}$	$L_{RE}^{(1,3)}$ = (1.97, 2.42) m	$U_{RE}^{(1,3)}$ = (11.06, 11.51) m	l = .40 m

Table 3.2: The parameters used to construct each Bad Set (Section 3.3.1) are provided for Experimental Configuration 2.

Bad Set Parameters for Configuration 3			
$B_M^{(1,2)}$	$L_M^{(1,2)}$ = (9.86, 3.08) m	$U_M^{(1,2)}$ = (10.76, 3.98) m	n/a
$B_M^{(1,3)}$	$L_M^{(1,3)}$ = (1.97, 2.42) m	$U_M^{(1,3)}$ = (2.87, 3.32) m	n/a
$B_{RE}^{1,2}$	$L_{RE}^{(1,2)}$ = (8.96, 2.18) m	$U_{RE}^{(1,2)}$ = (0.87, 5.2) m	l = .40 m
$B_{RE}^{(1,3)}$	$L_{RE}^{(1,3)}$ = (1.97, 2.42) m	$U_{RE}^{(1,3)}$ = (8.72, 9.17) m	l = .40 m

Table 3.3: The parameters used to construct each Bad Set (Section 3.3.1) are provided for Experimental Configuration 3.

figurations. This allows us to explore

Roundabout Drill

The roundabout drill layout is shown in Figure 3.1(a), where each path length is given by

$$D^1 = 11.62 \text{ m}, \ D^2 = 5.91 \text{ m and } D^3 = 14.22 \text{ m}.$$

A coordinate system is generated along each path, where the path origins are shown in Figure 3.1. Three bad set configurations are considered. The parameters used to construct the bad sets for Configuration 1 are given in Table 3.1, Configuration 2 are given in Table 3.2 and Configuration 2 are given in Table 3.3. The vehicles position of Figure 3.1(7) is represented in the position spaces $X_1^{1,2}$ and $X_1^{1,3}$ in Figure 3.1(b)-(c).

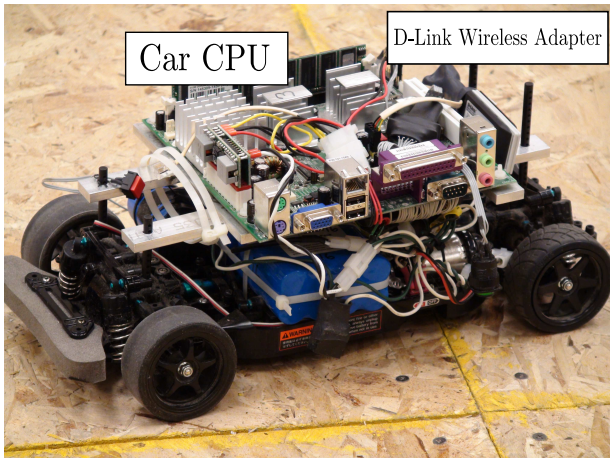


Figure 3.9: Custom Dynamic Vehicle.

Vehicle Hardware Description

Each vehicle is custom designed with a commercial RC chassis and standard electric motor, as seen in Figure 3.9. The dimensions of the vehicle are 0.17 m wide, and 0.38 m long. The on-board computer system consists of VIA EPIA Mini-ITX with a 600 MHz processor, 512 MB of RAM, and a 40 GB hard drive, as shown in Figure 3.9. The control and communication algorithms are written in C and run on the Mini-ITX using a Linux Fedora Core 5 operating system. Torque commands from control algorithms on the Mini-ITX are transmitted to a BrainStem Moto 1.0 motor controller via a serial connection, where torque is translated into voltage for the DC motor through motor maps [97].

Parameters for Longitudinal Dynamics

The parameters a and b that define the longitudinal dynamics of each vehicle, as defined in Section 3.2.1, were found through standard least squares techniques as:

$$\text{Vehicle 1:} \quad a = 3.77,\ b = -55;$$

$$\text{Vehicle 2:} \quad a = 5.07,\ b = -12;$$

$$\text{Vehicle 3:} \quad a = 6.43,\ b = -133;$$

where a is in $kg^{-1}mm^{-1}$, and b is in mm/s^2.

The maximum and minimum acceleration γ, as defined in Figure 3.2, are limited on each vehicle by $\gamma_{min} = -350mm/s^2$ and $\gamma_{max} = 350\ mm/s^2$. This implies the input sets are given by

$$\begin{aligned} U^1 &= [-79,\ 71]\ N \cdot mm; \\ U^2 &= [-67,\ 71]\ N \cdot mm; \\ U^3 &= [-34,\ 75]\ N \cdot mm. \end{aligned}$$

Communication and Measurement

Each vehicle is equipped with a wireless network card so that communication between the overhead positioning system, the vehicles, and the lab computers is achieved through a local 802.11b wireless network.

The overhead positioning system consists of four grey-scale cameras fixed to the laboratory ceiling and linked to two desktop computers via a FireWire cable. The computers use template matching to locate the patterns shown in Figure 3.11, which are affixed to the top of the vehicles. Each individual pattern is unique, which allows the system to distinguish individual vehicles. The patterns are circular, making the templates rotationally invariant so that orientation does not impede vehicle identification. The orientation is derived by calculating the angle of a line segment at the front of the template with respect to a reference angle. The desktop computers calculate the position and orientation of each vehicle, and over the wireless network send each vehicle *only* its individual position and orientation. The vehicle position is accurate within +/-50 mm and the orientation is accurate within +/-10 degrees. The vehicle velocity is calculated on-board each vehicle by differentiating and low pass filtering the positioning data.

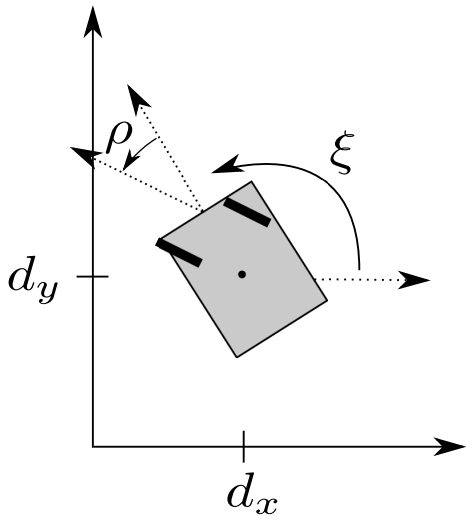


Figure 3.10: Kinematic model describing the vehicle dynamics.

Path Following Algorithm

A low level path following controller is implemented that keeps each vehicle on its respective path. A kinematic bicycle model (Figure 3.10) to describe the vehicle dynamics is given by

$$\dot{d}_x = v\cos(\xi + \rho),\ \dot{d}_y = v\sin(\xi + \rho),\ \dot{\xi} = \frac{v}{l}sin(\rho),$$

where d_x and d_y are the vehicle coordinates, ξ is the heading angle of the vehicle, ρ is the steering wheel offset angle, v is the absolute speed and l is the vehicle length.

At every instant of time, the vehicle determines the closest forward point within a waypoint list and uses a proportional controller to adjust the heading toward the target point by means of actuating the steering angle. Let the target point be given by χ_{tar}, then the target angle is given by $\xi_{tar}[k] = \arctan(\frac{\chi_{y,tar} - d_y[k]}{\chi_{x,tar} - d_x[k]})$, which a proportional controller uses to actuate the steering angle by $\rho[k] = K_P(\xi_{tar}[k] - \xi[k])$, where $\xi[k]$ represents the current heading.

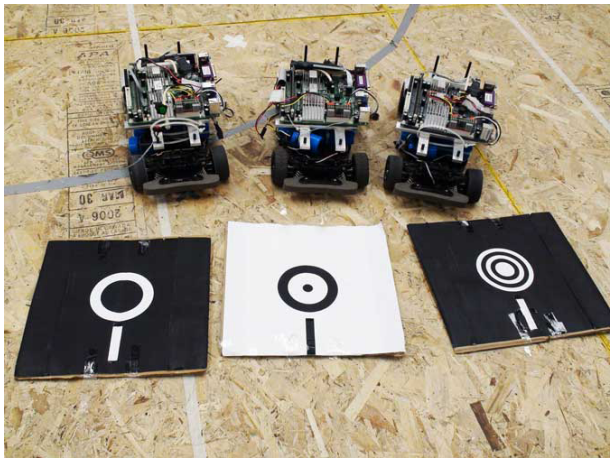


Figure 3.11: Template patterns used by the Positioning System for Vehicle Identification and Orientation.

3.6.2 Safety Controller Algorithm Implementation

This section examines how the control primitives are implemented. A block diagram that represents the logic of the software is depicted in Figure 3.12. We use a discrete time implementation of the safety control map $\mathbf{g}$ with sampling period $\Delta T = .1$ s. We start by discuss some details pertaining to how the algorithms from Section 3.5 are implemented.

Velocity Uncertainty

Both the modules $\mathbf{g}_M^k$ and $\mathbf{g}_{RE}^k$ require checking membership of the current state in the boundary of a restricted capture set. As discussed in Section 3.5, membership within a restricted capture set can be checked by computing a slice of the capture set corresponding to the current velocity. To accommodate velocity error arising from noisy position measurements, we show how to compute a slice of the capture set for a set of velocities.

Model the velocity error at step k as the set $\mathcal{X}_2[k] := [x_2[k] - 50, x_2[k] + 50]$ given the velocity measurement $x_2[k]$. From the definition of $\mathbf{g}$, control is only applied when the state $x[k]$ lies on the boundary of a restricted capture set, as defined in Section 3.3.4. We

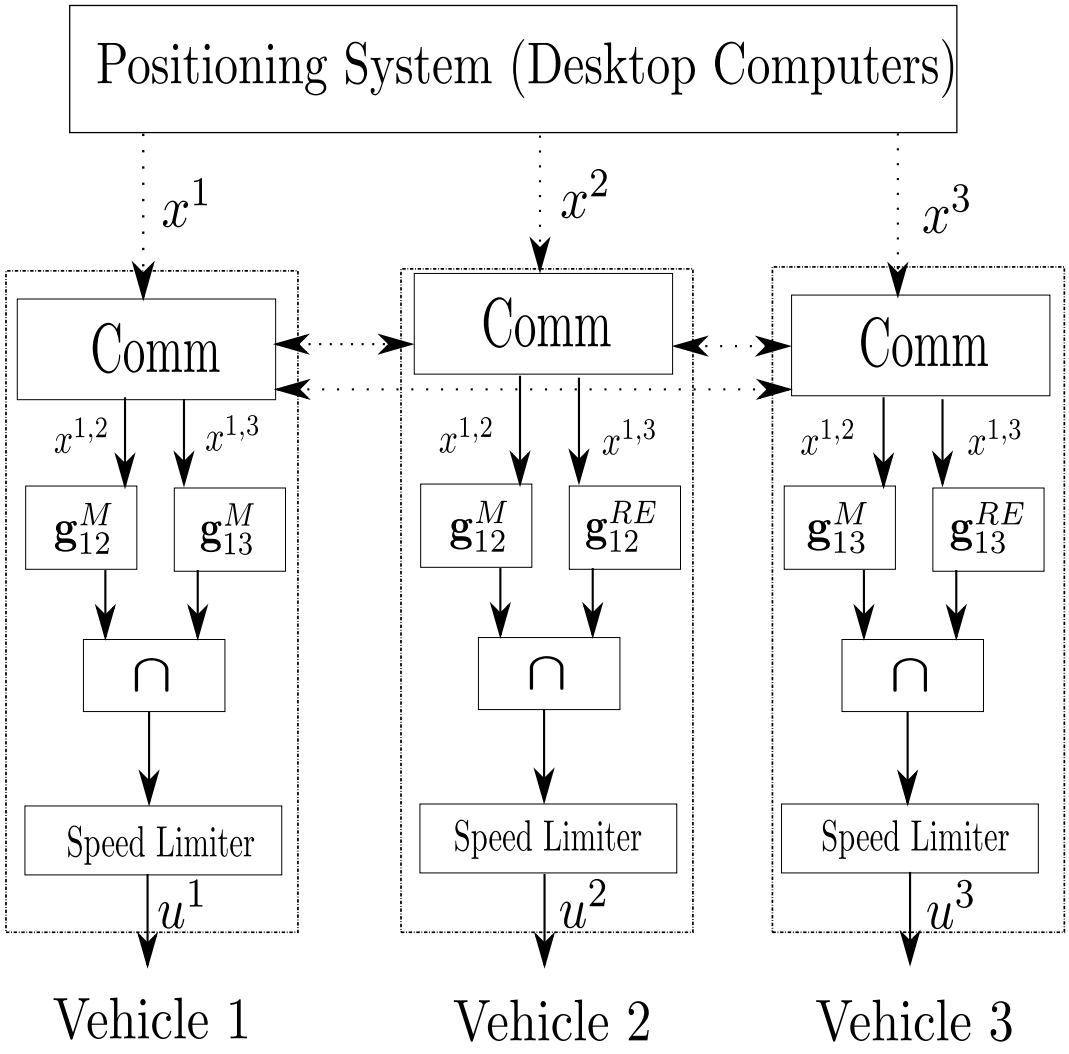


Figure 3.12: High-level software architecture depicting the interaction of the control components of the entire system. Signals sent over the air are denoted by dashed arrows. The speed limiter block keeps the speed within the interval $[v_{min}, v_{max}]$.

check if the state belongs to the boundary of the restricted capture set $C(\mathbf{u})$ as follows:

$$x[k] \in \partial C(\mathbf{u}) \Leftrightarrow x[k] \notin C(\mathbf{u}) \text{ and } x[k+1] \in C(\mathbf{u}). \tag{3.23}$$

When computing $x[k+1]$ with an imperfect velocity measurement $\mathcal{X}_2[k]$, we must consider *all* possible future positions. Define the set of future positions $\mathcal{X}_1[k+1] := x_1[k]+\Delta T\mathcal{X}_2[k]$. Determining if any future state within $\mathcal{X}[k+1]$ is in the restricted capture set $C(\mathbf{u})$ requires checking whether or not $\mathcal{X}[k+1] \cap C(\mathbf{u}) \neq \emptyset$. Therefore, we check if the state belongs to

the boundary of the restricted capture set $C(\mathbf{u})$ as follows:

$$x[k] \in \partial C(\mathbf{u}) \Leftrightarrow x[k] \notin C(\mathbf{u}) \neq \emptyset \text{ and } \mathcal{X}[k+1] \cap C(\mathbf{u}) \neq \emptyset. \tag{3.24}$$

Under the order preserving property of the dynamics with respect to the state, intersection of a set of states with a restricted capture set $C(\mathbf{u})$ can be found by checking whether for $j \in \mathscr{I}$ there exists $m \in \mathbb{N}$ such that

$$\mathcal{X}^j[k+1] \cap]L^j(\sup \mathcal{X}_2^j[k], \mathbf{u}_m^j), U^j(\inf \mathcal{X}_2^j[k], \mathbf{u}^j)_m[\neq \emptyset.$$

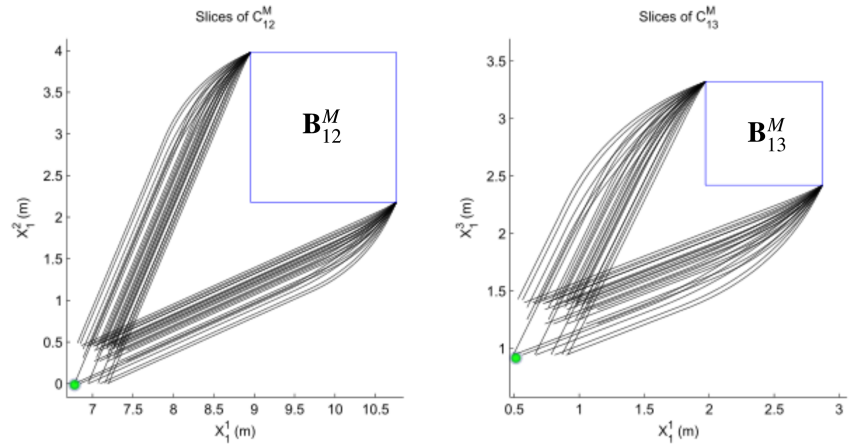
3.6.3 Interaction of Modules

We check whether or not the composition of the modules is non-conflicting. We first check that the merging modules are non-conflicting, which follows if $\mathbf{C}_M^{(1,2)} \cap \mathbf{C}_M^{(1,3)} =$. We next check that the rear-end modules and the merging modules do not conflict each other, which is the case when $\partial \mathbf{A}_{RE}^k \cap \mathbf{C}_{RE}^k \subset \mathbf{B}_M^k$ for $k \in \mathscr{I}$. To check if $\mathbf{C}_M^{(1,2)} \cap \mathbf{C}_M^{(1,3)} =$, we compute the maximum size of the capture set $C_M^{(1,2)}$, which is found by computing capture set slices for all possible velocity values $x_2^{(1,2)} \in X_2^{(1,2)}$. We similarly compute the maximum size of the capture set $C_M^{(1,3)}$ by computing capture set slices for all possible velocity values $x_2^{(1,3)} \in X_2^{(1,3)}$. Each capture set slice is projected onto the position space in Figure 3.13.

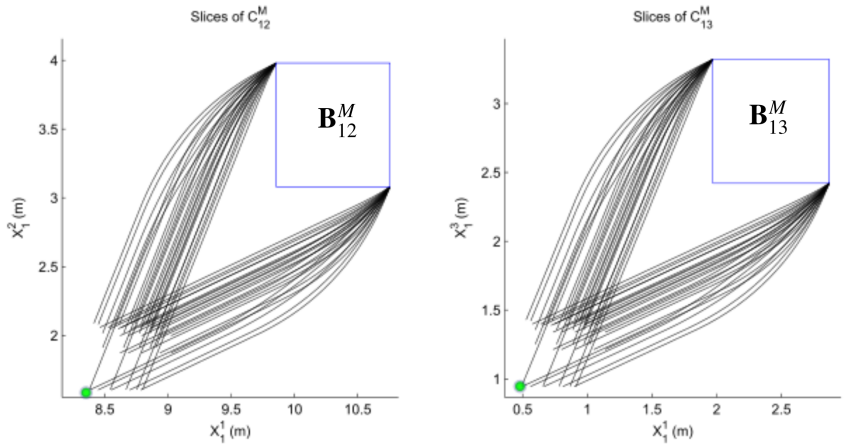
From Figure 3.13(a), we see that $\alpha_{12} = (6.78m, 0.0m)$ and $\alpha_{13} = (0.49m, 0.94m)$ for Configurations 1 and 2. From Figure 3.13(b), we see that $\alpha_{12} = (8.37m, 1.60m)$ and $\alpha_{13} = (0.49m, 0.94m)$ for Configuration 3. In Figure 3.14(a), we project the merging capture sets onto the roundabout drill to show that the capture sets do not overlap for all configurations.

Hence, we have that $\mathbf{C}_M^{(1,2)} \cap \mathbf{C}_M^{(1,3)} = \emptyset$ for all configurations, implying that $\mathbf{g}_M^{(1,2)}(x) \cap \mathbf{g}_M^{(1,3)}(x) \neq \emptyset$ for all configurations.

We next verify that the rear-end modules never interfere with the merging modules. We generate all possible rear-end capture set slices for every velocity $x_2^{1,2} \in X_2^{1,2}$ in Figure



(a)



(b)

Figure 3.13: All possible Merging Capture Set Slices for (a) Configurations 1 and 2, (b) Configuration 3.

3.15(a) for Configuration 1, Figure 3.15(b) for Configuration 2, and Figure 3.15(c) for Configuration 3. We see that we have met the Conditions that $C^{(1,2)}_{RE} \cap \partial\mathbf{A}^{(1,2)}_{RE} \subset \mathbf{B}^{(1,2)}_{M}$ and that $C^{(1,3)}_{RE} \cap \partial\mathbf{A}^{(1,3)}_{RE} \subset \mathbf{B}^{(1,3)}_{M}$, sufficient conditions guaranteeing we never have $\mathbf{g}^{(1,2)}_{M}(x) \cap \mathbf{g}^{(1,2)}_{RE}(x) = \emptyset$ or $\mathbf{g}^{(1,3)}_{M}(x) \cap \mathbf{g}^{(1,3)}_{RE}(x) = \emptyset$.

3.6.4 Experimental Results

We provide experimental results for ten separate trials. Each trial starts by placing the vehicles in a configuration outside of all capture sets, and then running each of them at a velocity set-point of .7 m/s. The duration of each trial is arbitrary, ranging from 34 seconds

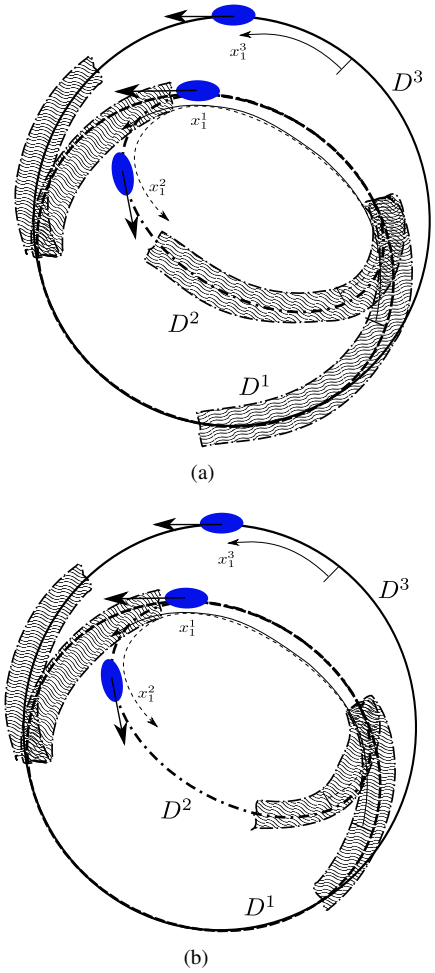
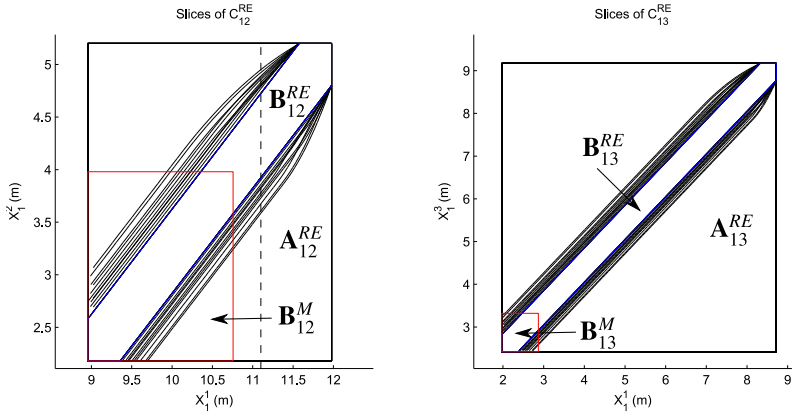


Figure 3.14: Merging Capture Sets Projected onto Roundabout Drill for (a) Configurations 1 and 2, (b) Configuration 3.

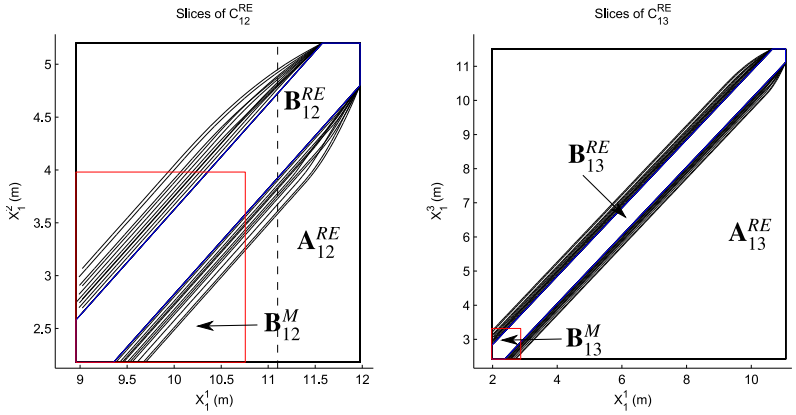
to 228 seconds. If no modules are active at a given instant, the user can change the velocity set-point of each vehicle within the interval $[v_{min}, v_{max}]$ via the desktop computers. Of the ten trials, six trials were performed with Configuration 1 (Table 3.1), two trials were performed with Configuration 2 (Table 3.2) and two trials were performed with Configuration 3 (Table 3.3).

Overview

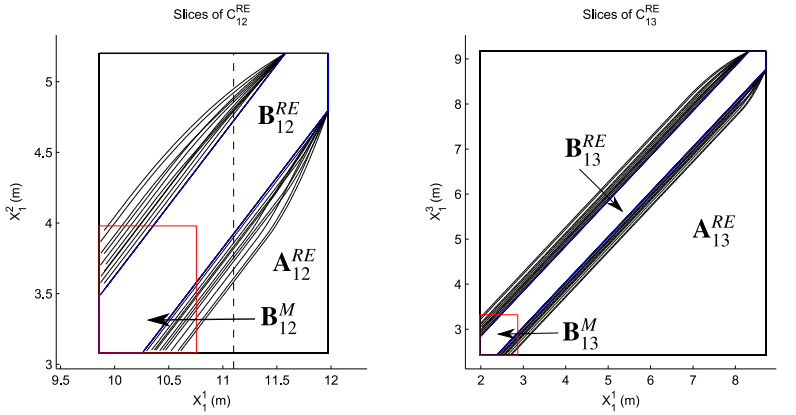
A table containing the number of times each module activated during each trial is provided in Table 3.4. We plot the complete history of the flow over all time projected onto the



(a)



(b)



(c)

Figure 3.15: All Rear-End Capture Set slices shown for (a) Configuration 1, (b) Configuration 2, and (c) Configuration 3. In each figure, the bad set $\mathbf{B}_M^k$ is shown in red, the bad set $\mathbf{B}_{RE}^k$ is shown in blue, and the enable set $\mathbf{A}_{RE}^k$ is shown in black.

position spaces $X_1^{(1,2)}$ and $X_1^{(1,3)}$ in Figure 3.16, which are separated by Configuration. The safety specification is maintained if the flow, as projected into these spaces, never enters

Experimental Results						
Trial	Time (sec)	Configuration	Merging (12)	Rear-End (12)	Merging (13)	Rear-End (13)
1	159.9	1	4	0	2	3
2	228.0	1	5	0	1	5
3	158.0	1	7	0	1	3
4	131.8	1	5	0	0	5
5	203.5	1	10	0	0	4
6	191.2	1	9	0	1	3
7	177.0	2	4	0	0	2
8	210.0	2	8	1	0	5
9	34.2	3	0	0	1	1
10	112.7	3	0	0	2	1
Total	112.7	3	0	0	2	1

Table 3.4: Statistics for all Trials.

Experiment Numbers			
Trial Number	Coupling	Entered Capture Set	Entered Bad Set
1	0	0	0
2	0	4	0
3	1	1	1
4	2	0	0
5	0	1	0
6	0	3	0
7	2	5	0
8	3	3	0
9	0	0	0
10	0	0	2

Table 3.5: Statistics for all Trials (cont).

any of the bad sets.

We next provide a detailed example of the execution of the module $\mathbf{g}_M^{(1,3)}$ for a specific instance, shown in Figure 3.17. The evolution of the flow in the state space along with the control signals applied are provided in Figure 3.20.

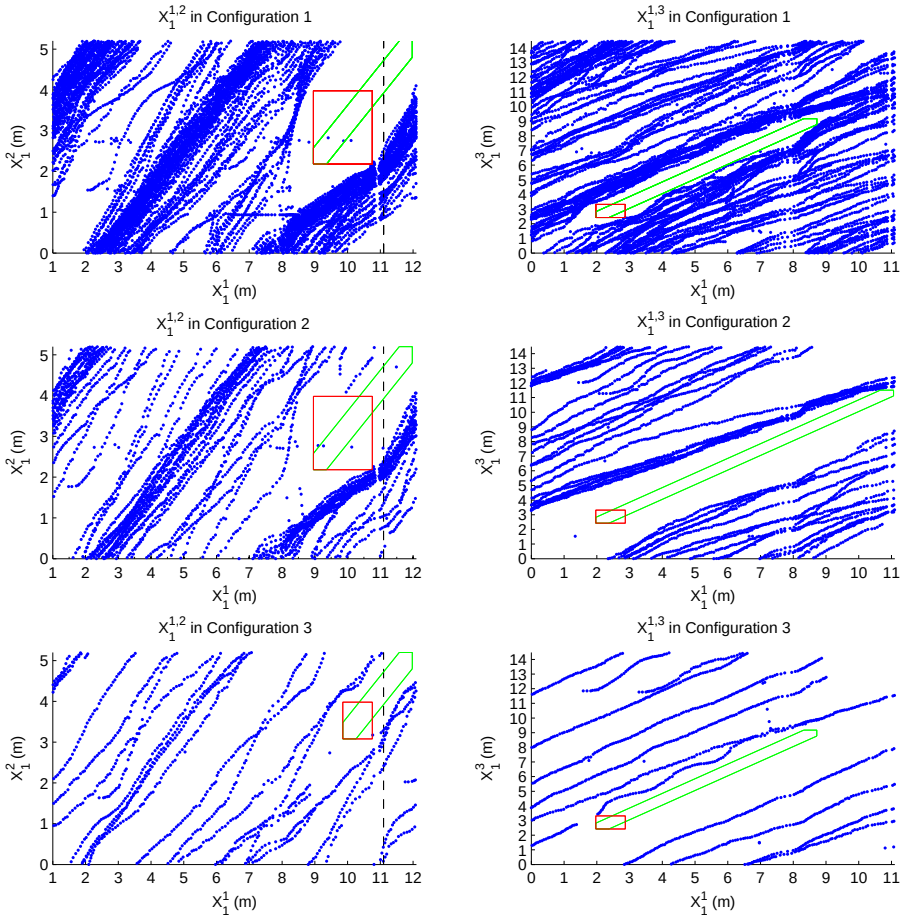


Figure 3.16: Evolution of the state $x_1^{(1,2)}$ in the left column and the state $x_1^{(1,3)}$ in the right column. The red box denotes $\Omega_M^{(1,2)}$ and the green set denotes $\Omega_{RE}^{(1,3)}$. Trials 1-6 are shown in the plots of Configuration 1. Trials 7 and 8 are shown in the plots of Configuration 2. Trials 9 and 10 are shown in the plots of Configuration 3

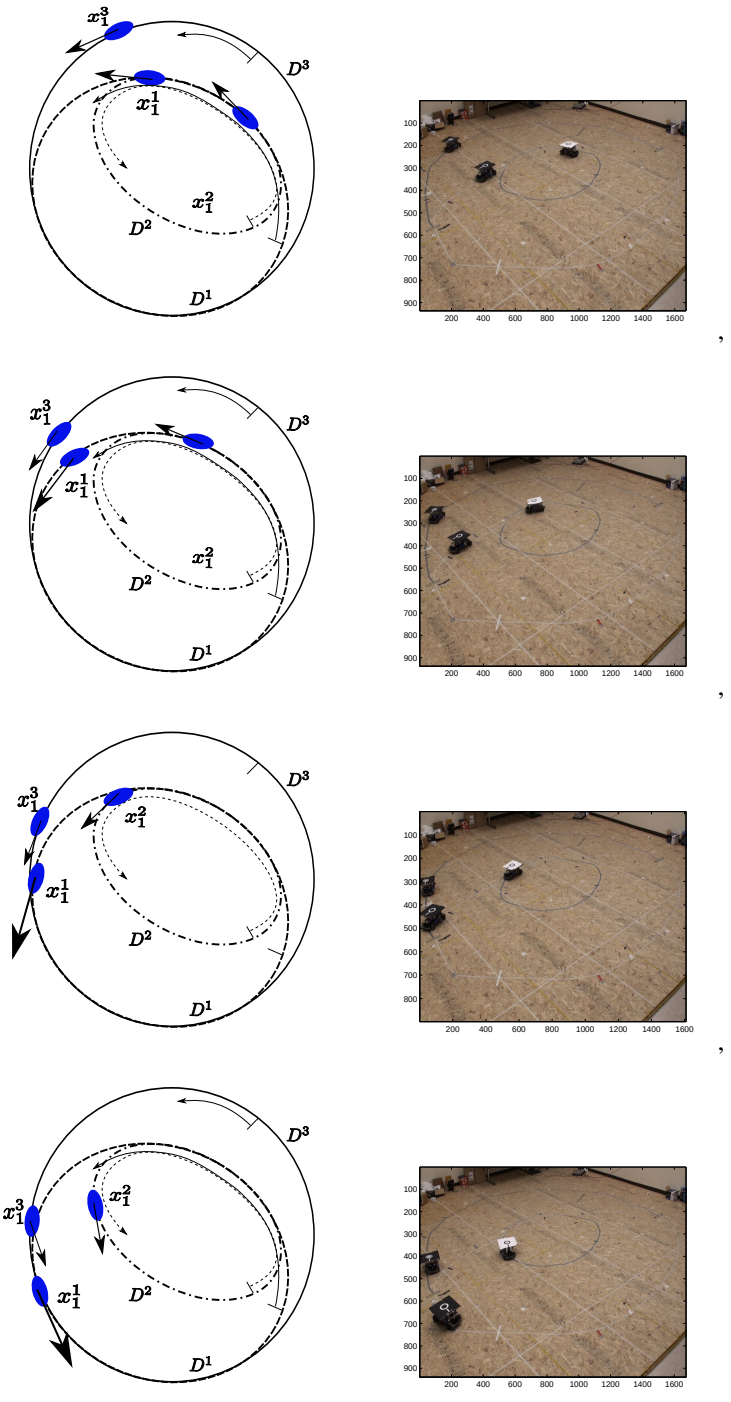


Figure 3.17: Resolution for $g_M^{(1,3)}$ below the Bad Set $B_M^{(1,3)}$ with rows depicting vehicle trajectories with corresponding test images. System flow is shown in Figure 3.20.

3.6.5 Conflict Resolution Scenarios

We now present individual instances of the activation of each module in all the possible ways. In each case, a Figure with snaps shots of the evolution is given with the state, bad set and capture set slice shown. In addition, the control signals applied along with vehicle velocities are shown.

Merging module between vehicle 1 and vehicle 2, vehicle 3 is free

We present instances of the $\bar{g}_M^{(1,2)}(\hat{x})_{coop}$ module working for two conflict resolutions:

(i) Vehicle 1 = u_H^1, Vehicle 2 = u_L^2, Vehicle 3 = anything;

(ii) Vehicle 1 = u_L^1, Vehicle 1 = u_H^2, Vehicle 3 = anything.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,2)}$ position space, the flow $\phi^{(1,2)}$ proceeds below the bad set $B_M^{(1,2)}$. This is shown in Figure 3.18.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,2)}$ position space, the flow $\phi^{(1,2)}$ proceeds above the bad set $B_M^{(1,2)}$. This is shown in Figure 3.19.

Merging module between vehicle 1 and vehicle 3, vehicle 2 is free

We present instances of the $\bar{g}_M^{(1,3)}(\hat{x})_{coop}$ module working for two conflict resolutions:

(i) Vehicle 1 = u_H^1, Vehicle 2 = free, Vehicle 3 = u_L^3;

(ii) Vehicle 1 = u_L^1, Vehicle 1 = free, Vehicle 3 = u_H^3.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}$ proceeds below the bad set $B_M^{(1,3)}$. This is shown in Figure 3.20.

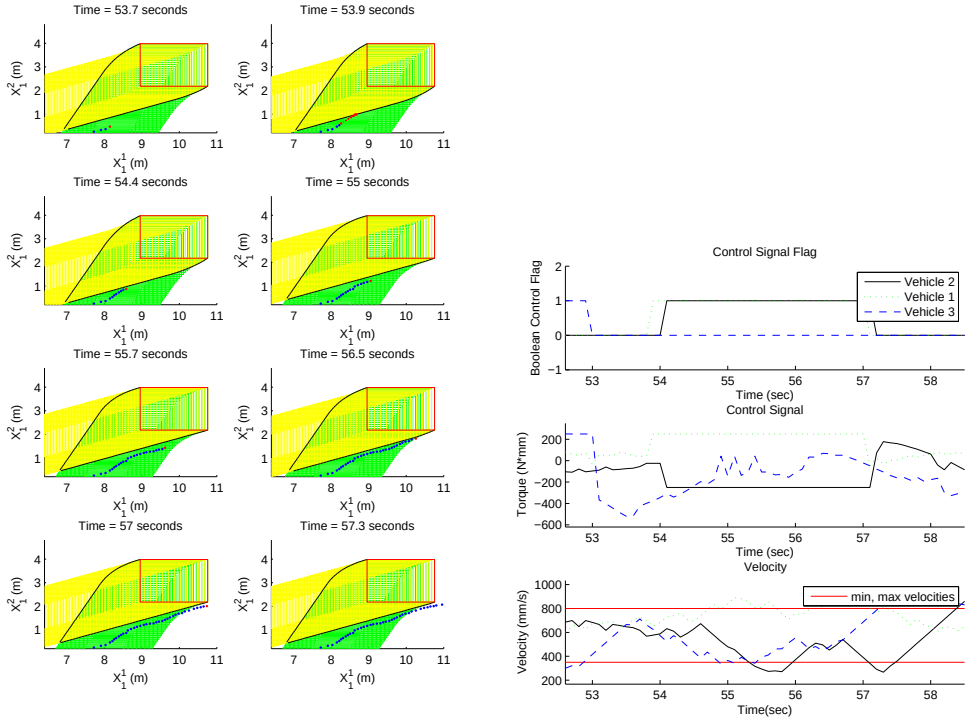


Figure 3.18: The above plots depict snap shots of the dynamic evolution of the system for which only the cooperative module $\bar{g}_M^{(1,2)}(x)$ is activated. The vehicle not activated by any module applies velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,2}$, with the red dot corresponding to $\phi(t, x, \mathbf{u})$ at the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}_M^{(1,2)}$ as projected onto the space of constraint $X_1^{(1,2)}$. The yellow set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ corresponding to the current speed and the green set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{L}})$ corresponding to the current speed. Plots of the control module evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}$ proceeds above the bad set $B_M^{(1,3)}$. This is shown in Figure 3.21.

Rear-End module between vehicle 1 and vehicle 2, vehicle 3 is free

We present instances of the $\bar{g}_{RE}^{(1,2)}(\hat{x})_\delta$ module working for two conflict resolutions:

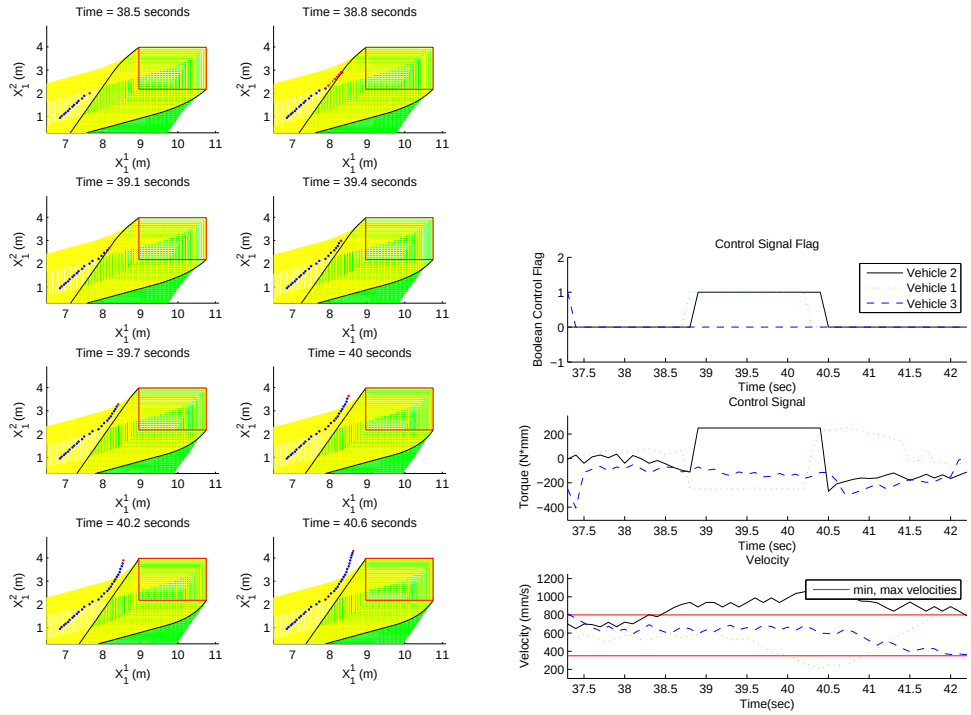


Figure 3.19: The above plots depict snap shots of the dynamic evolution of the system for which only the cooperative module $\bar{g}_M^{(1,2)}(x)$ is activated. The vehicle not activated by any module applies velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,2}$, with the red dot corresponding to $\phi(t, x, \mathbf{u})$ at the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}_M^{(1,2)}$ as projected onto the space of constraint $X_1^{(1,2)}$. The yellow set represents the slice of the restricted capture set $C_M^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ corresponding to the current speed and the green set represents the slice of the restricted capture set $C_M^{(1,3)}(\mathbf{u}_{\mathcal{L}})$ corresponding to the current speed. Plots of the control module evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

(i) Vehicle 1 = free, Vehicle 2 = u_L^2, Vehicle 3 = free;

(ii) Vehicle 1 = free, Vehicle 2 = u_H^2, Vehicle 3 = free.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{1,2}$ position space, the flow $\phi^{1,2}(t)$ proceeds above the bad set $B_{RE}^{1,2}$. Plotted as Figure 3.22.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,2)}$ position space, the flow $\phi^{1,2}(t)$ proceeds below the bad set $B_{RE}^{(1,2)}$. Plotted as Figure 3.23.

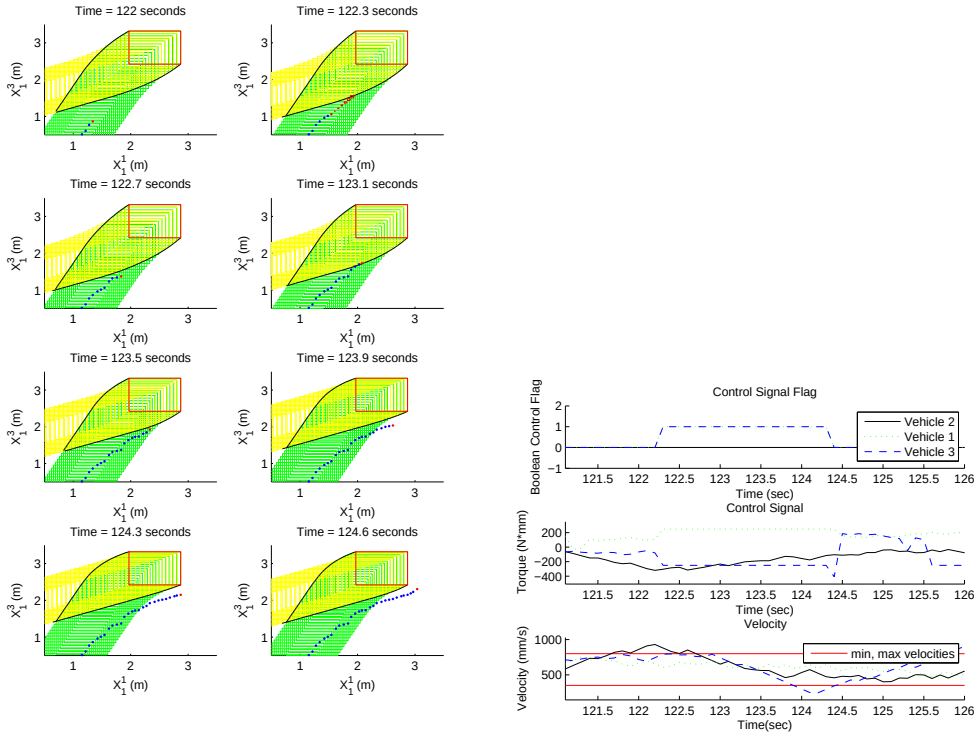


Figure 3.20: The above plots depict snap shots of the dynamic evolution of the system for which only the cooperative module $\bar{g}_M^{(1,3)}(x)$ is activated. The vehicle not activated by any module applies velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,3}$, with the red dot corresponding to $\phi(t, x, \mathbf{u})$ at the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}_M^{(1,3)}$ as projected onto the space of constraint $X_1^{(1,3)}$. The yellow set represents the slice of the restricted capture set $C_M^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ corresponding to the current speed and the green set represents the slice of the restricted capture set $C_M^{(1,3)}(\mathbf{u}_{\mathcal{L}})$ corresponding to the current speed. Plots of the control module evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

Rear-End module between vehicle 1 and vehicle 3, vehicle 2 is free

We present instances of the $\bar{g}_{RE}^{(1,3)}(\hat{x})_\delta$ module working for two conflict resolutions:

(i) Vehicle 1 = free, Vehicle 2 = free, Vehicle 3 = u_L^3;

(ii) Vehicle 1 = free, Vehicle 1 = free, Vehicle 3 = u_H^3.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}(t)$ proceeds below the bad set $B_{RE}^{(1,3)}$. Plotted as Figure 3.24.

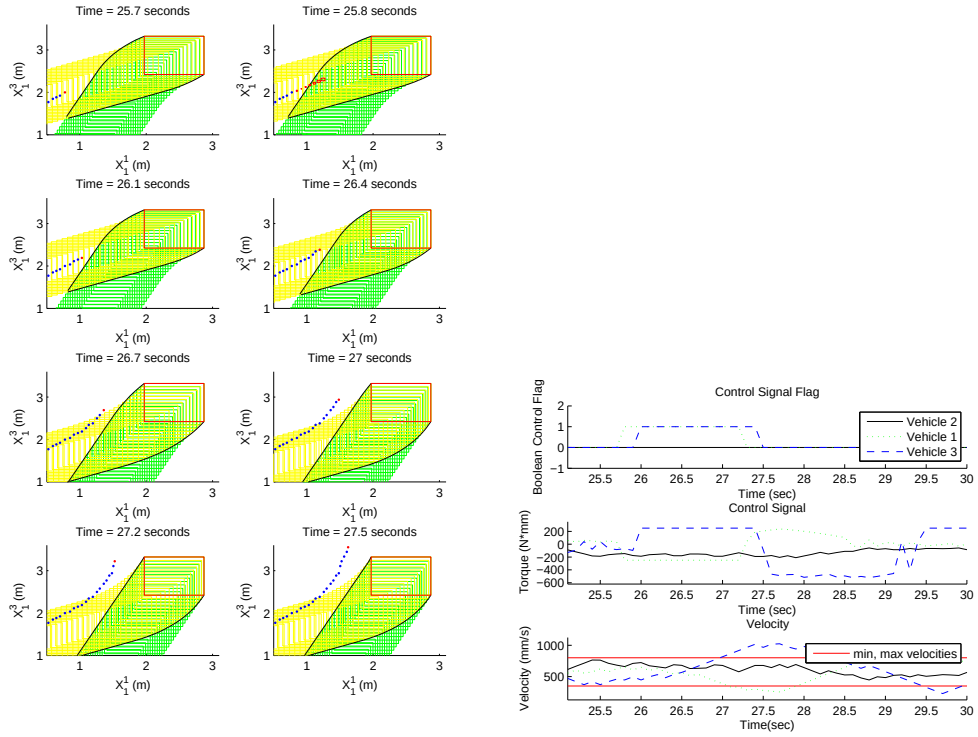


Figure 3.21: The above plots depict snap shots of the dynamic evolution of the system for which only the cooperative module $\bar{g}_M^{(1,3)}(x)$ is activated. The vehicle not activated by any module applies velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t,x,\mathbf{u})$ projected onto the space of constraint $X_1^{1,3}$, with the red dot corresponding to $\phi(t,x,\mathbf{u})$ at the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}_M^{(1,3)}$ as projected onto the space of constraint $X_1^{(1,3)}$. The yellow set represents the slice of the restricted capture set $C_M^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ corresponding to the current speed and the green set represents the slice of the restricted capture set $C_M^{(1,3)}(\mathbf{u}_{\mathcal{L}})$ corresponding to the current speed. Plots of the control module evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

One instance, hit boundary and show flow evolving on boundary and passing boundary. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}(t)$ proceeds above the bad set $B_{RE}^{(1,3)}$. Plotted as Figure 3.25.

Merging module between vehicle 1 and vehicle 2, rear-end module between vehicle 1 and vehicle 3

We present instances of the $\bar{g}_M^{(1,2)}(\hat{x})_{coop} \cap \bar{g}_{RE}^{(1,3)}(\hat{x})_{\delta}$ module working for four conflict resolutions:

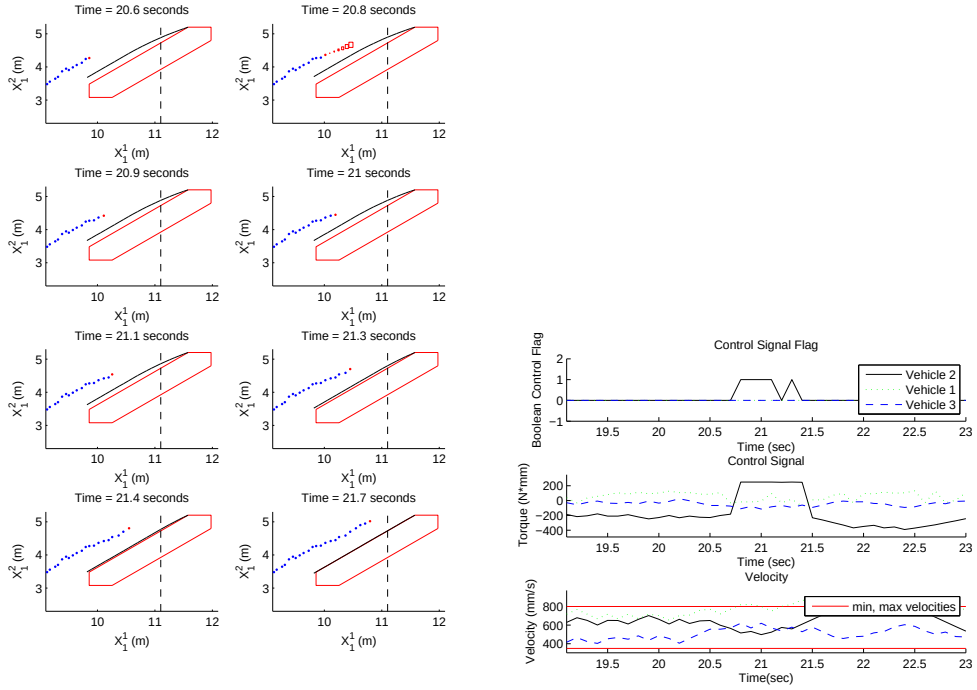


Figure 3.22: The above plots depict snap shots of the dynamic evolution of the system for which only the competitive module $\bar{g}^{1,2}_{RE}(x)$ is activated. The vehicles not activated by any modules apply velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,2}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}^{(1,3)}_{RE}$ as projected onto the space of constraint $X_1^{1,2}$. The upper boundary for the slice of the capture set $C^{1,2}_{RE}$ corresponding to the current speed is shown as a black line. Vehicle 1 never applies any brake control as the module $\bar{g}^{(1,3)}_{M}$ never activates, therefore the flow does not stay on the boundary of the capture set $C^{1,2}_{RE}$. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

(i) Vehicle 1 = u^1_H, Vehicle 2 = u^2_L, Vehicle 3 = u^3_L;

(ii) Vehicle 1 = u^1_L, Vehicle 2 = u^2_H, Vehicle 3 = u^3_L;

(iii) Vehicle 1 = u^1_H, Vehicle 2 = u^2_L, Vehicle 3 = u^3_H;

(iv) Vehicle 1 = u^1_L, Vehicle 2 = u^2_H, Vehicle 3 = u^3_H.

One instance, hit boundary and show flow evolving on boundary and passing boundary for both simultaneously. In $X_1^{(1,2)}$ position space, the flow $\phi^{(1,2)}(t)$ proceeds below the bad set $B^{(1,2)}_M$. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}(t)$ proceeds below the bad set $B^{(1,3)}_{RE}$. Plotted as Figure 3.26.

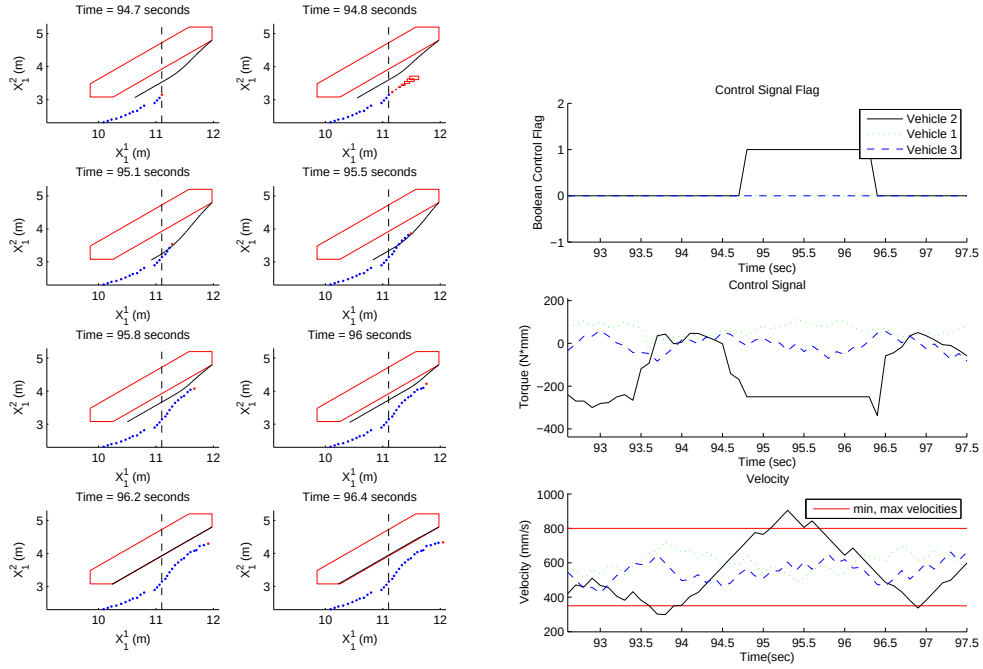


Figure 3.23: The above plots depict snap shots of the dynamic evolution of the system for which only the competitive module $\bar{g}^{1,2}_{RE}(x)$ is activated. The vehicles not activated by any modules apply velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,2}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}^{(1,3)}_{RE}$ as projected onto the space of constraint $X_1^{1,2}$. The Lower boundary for the slice of the capture set $C^{1,2}_{RE}$ corresponding to the current speed is shown as a black line. Vehicle 1 never applies any brake control as the module $\bar{g}^{(1,3)}_{M}$ never activates, therefore the flow does not stay on the boundary of the capture set $C^{1,2}_{RE}$. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

One instance, hit boundary and show flow evolving on (off) boundary and passing boundary for both simultaneously. In $X_1^{(1,2)}$ position space, the flow $\phi^{(1,2)}(t)$ proceeds above the bad set $B^{(1,2)}_{M}$. In $X_1^{(1,2)}$ position space, the flow $\phi^{(1,3)}(t)$ proceeds below the bad set $B^{(1,3)}_{RE}$. Plotted as Figure 3.27.

One instance, hit boundary and show flow evolving on (off) boundary and passing boundary for both simultaneously. In $X_1^{1,2}$ position space, the flow $\phi^{(1,2)}(t)$ proceeds below the bad set $B^{(1,2)}_{M}$. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}(t)$ proceeds above the bad set $B^{(1,3)}_{RE}$. Plotted as Figure 3.28.

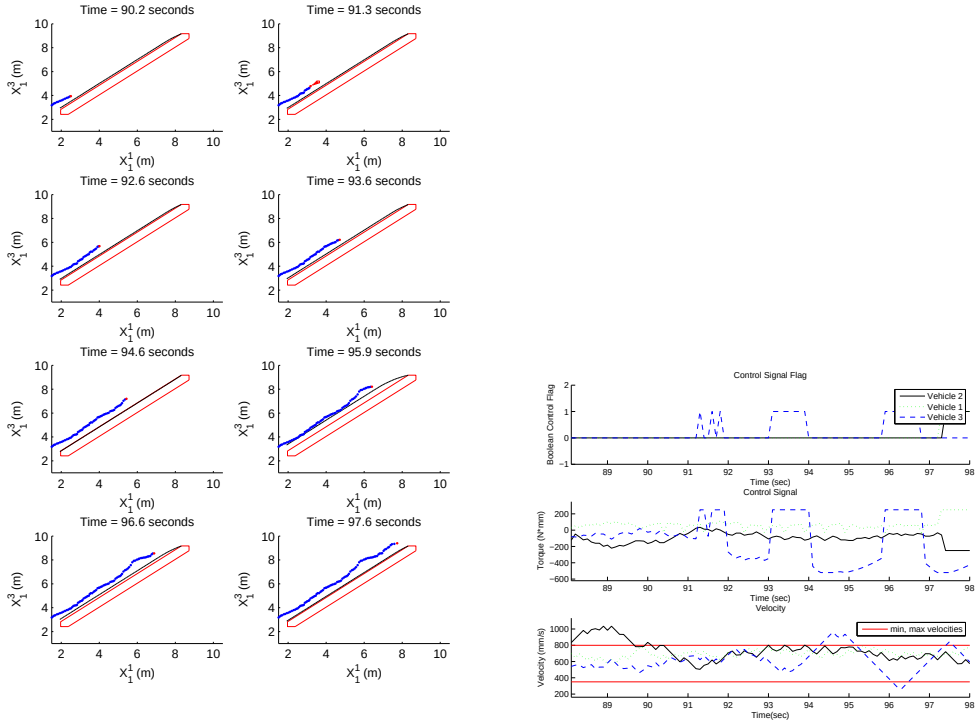


Figure 3.24: The above plots depict snap shots of the dynamic evolution of the system for which only the competitive module $\bar{g}_{RE}^{(1,3)}(x)$ is activated. The vehicles not activated by any modules apply velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,3}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}_{RE}^{(1,3)}$ as projected onto the space of constraint $X_1^{1,3}$. The upper boundary for the slice of the capture set $C_{RE}^{(1,3)}$ corresponding to the current speed is shown as a black line. Vehicle 1 never applies any brake control as the module $\bar{g}_M^{(1,2)}$ never activates, therefore the flow does not stay on the boundary of the capture set $C_{RE}^{(1,3)}$. Plots of the control evaluation, inputs applied, and velocties for each vehicle are depicted in the lower plots.

One instance, hit boundary and show flow evolving on boundary and passing boundary for both simultaneously. In $X_1^{(1,2)}$ position space, the flow $\phi^{(1,2)}(t)$ proceeds above the bad set $B_M^{(1,2)}$. In $X_1^{(1,3)}$ position space, the flow $\phi^{(1,3)}(t)$ proceeds above the bad set $B_{RE}^{(1,3)}$. Plotted as Figure 3.29.

3.7 Discussion

There were instances where the state entered the capture set during experimentation. Possible causes for this include actuator delays, communication delays, poor position infor-

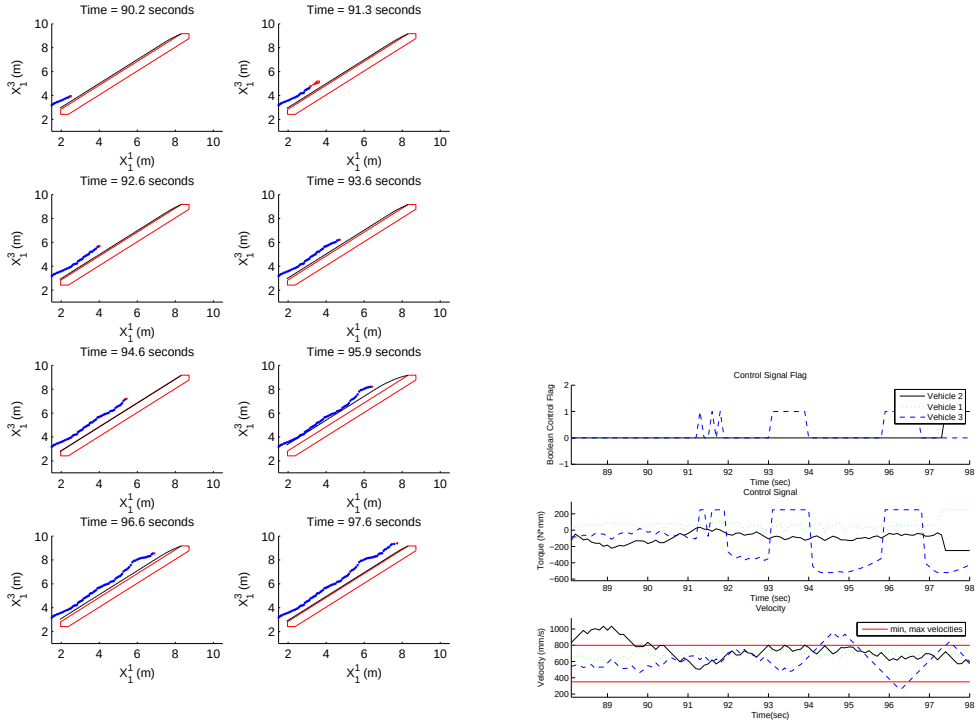


Figure 3.25: The above plots depict snap shots of the dynamic evolution of the system for which only the competitive module $\bar{g}_{RE}^{(1,3)}(x)$ is activated. The vehicles not activated by any modules apply velocity maintaining commands through user directed action. The top figure depicts 8 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{1,3}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red set depicts the bad set $\mathbf{B}_{RE}^{(1,3)}$ as projected onto the space of constraint $X_1^{1,3}$. The upper boundary for the slice of the capture set $C_{RE}^{(1,3)}$ corresponding to the current speed is shown as a black line. Vehicle 1 never applies any brake control as the module $\bar{g}_{M}^{(1,2)}$ never activates, therefore the flow does not stay on the boundary of the capture set $C_{RE}^{(1,3)}$. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

mation and path following errors.

3.7.1 Actuator Delays

It is assumed in theory that each vehicle can apply torque instantaneously as dictated by the controller. In practice, actuator delays often reach .5 sec, therefore, rather than checking if (3.23) holds for $x_1[k]$ and $\mathcal{X}_1[k+1]$, we instead check if (3.24) holds for six time steps into the future, that is for $\mathcal{X}_1[k+6]$.

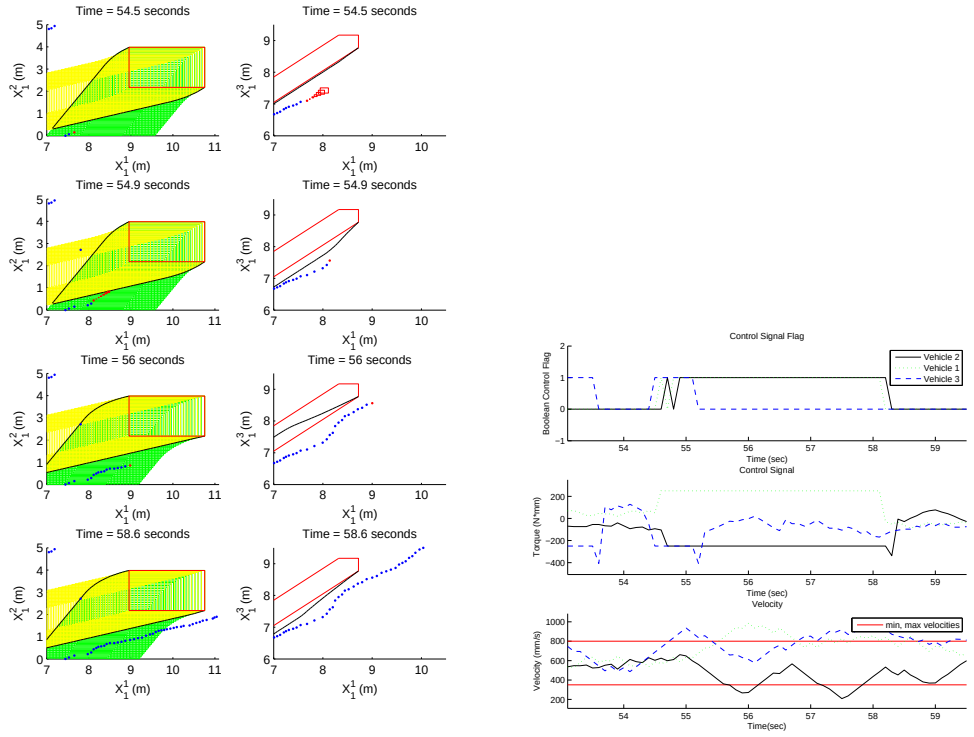


Figure 3.26: The above plots depict snap shots of the dynamic evolution of the system for which both the competitive module $\bar{g}_{RE}^{(1,3)}(x)$ and the cooperative module $\bar{g}_{M}^{(1,2)}(x)$ are activated. When not controlled by a module, the inputs are applied through user defined maintain velocity commands. The left column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,2)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The right column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,3)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red sets depict the bad sets $\mathbf{B}_{M}^{(1,2)}$ and $\mathbf{B}_{RE}^{(1,3)}$ as projected onto the relevant spaces of constraint $X_1^{(1,2)}$ and $X_1^{(1,3)}$. In the left column, for the current speed, the yellow set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ and the green set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{L}})$. In the right column, the lower boundary for the slice of the capture set $C_{RE}^{(1,3)}$ corresponding to the current speed is shown as a black line. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

3.7.2 Communication Delay

Ideally, the vehicles have access to the current state information from the other vehicles. However, there are periodic delays in the wireless network that result in the vehicles receiving either corrupted position data or not receiving anything. There are cases without updated state information where the flow can be thought to have entered the capture set,

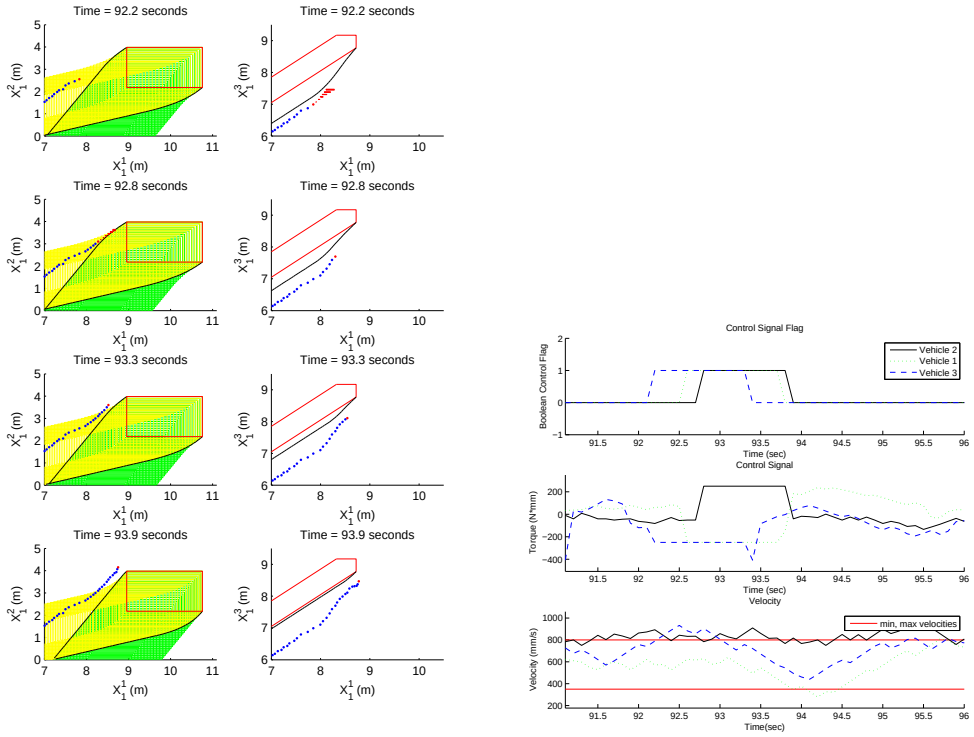


Figure 3.27: The above plots depict snap shots of the dynamic evolution of the system for which both the competitive module $\bar{g}_{RE}^{(1,3)}(x)$ and the cooperative module $\bar{g}_{M}^{(1,2)}(x)$ are activated. When not controlled by a module, the inputs are applied through user defined maintain velocity commands. The left column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,2)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The right column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,3)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red sets depict the bad sets $\mathbf{B}_{M}^{(1,2)}$ and $\mathbf{B}_{RE}^{(1,3)}$ as projected onto the relevant spaces of constraint $X_1^{(1,2)}$ and $X_1^{(1,3)}$. In the left column, for the current speed, the yellow set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ and the green set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{L}})$. In the right column, the lower boundary for the slice of the capture set $C_{RE}^{(1,3)}$ corresponding to the current speed is shown as a black line. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

corresponding to a false positive.

3.7.3 Severe Position Errors

There were observed cases of the position error suddenly jumping an order of a car length or more within a single time step. While there is the possibility of received state information jumping significantly after temporary communication loss, the magnitude of these

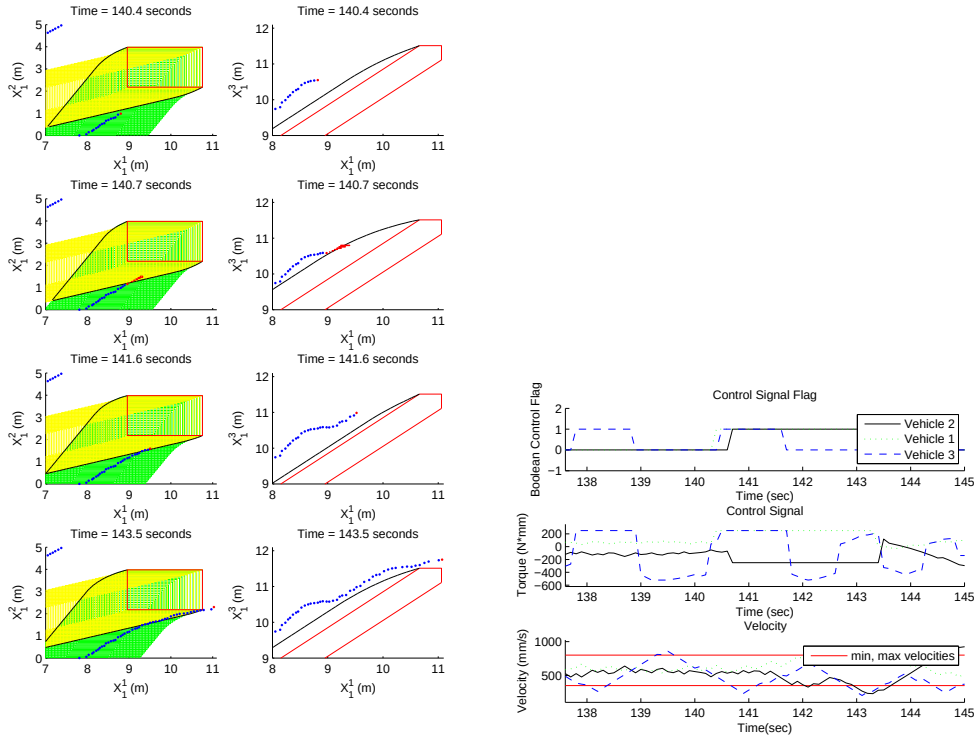


Figure 3.28: The above plots depict snap shots of the dynamic evolution of the system for which both the competitive module $\bar{g}_{RE}^{(1,3)}(x)$ and the cooperative module $\bar{g}_{M}^{(1,2)}(x)$ are activated. When not controlled by a module, the inputs are applied through user defined maintain velocity commands. The left column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,2)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The right column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,3)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red sets depict the bad sets $\mathbf{B}_{M}^{(1,2)}$ and $\mathbf{B}_{RE}^{(1,3)}$ as projected onto the relevant spaces of constraint $X_1^{(1,2)}$ and $X_1^{(1,3)}$. In the left column, for the current speed, the yellow set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ and the green set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{L}})$. In the right column, the lower boundary for the slice of the capture set $C_{RE}^{(1,3)}$ corresponding to the current speed is shown as a black line. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

errors signify a potential error in the positioning system much larger than the nominal error assumed.

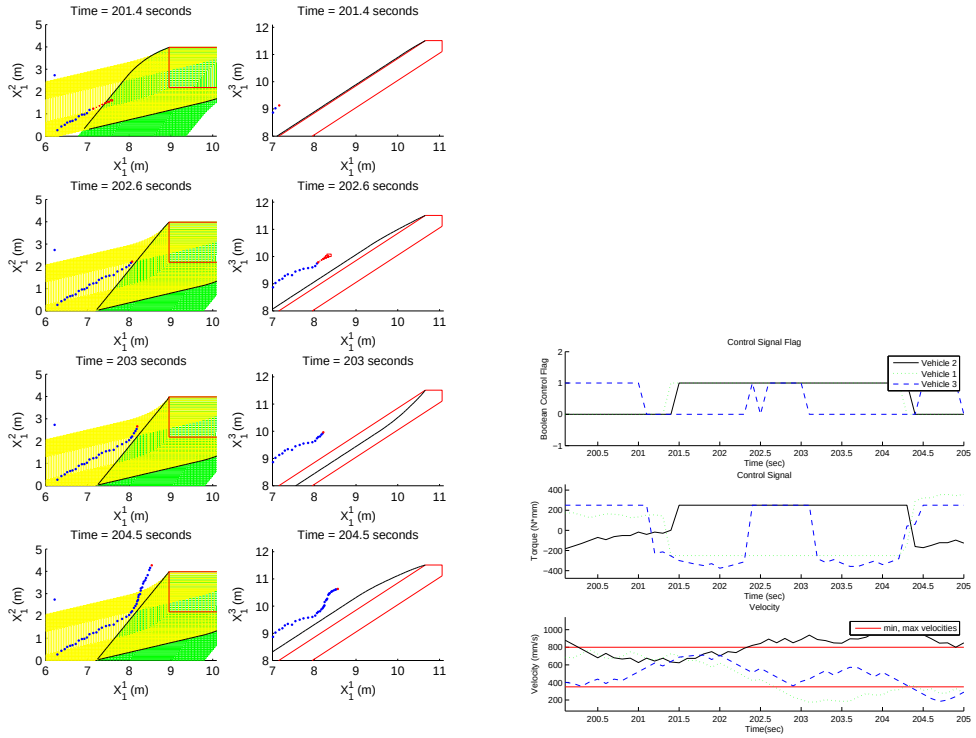


Figure 3.29: The above plots depict snap shots of the dynamic evolution of the system for which both the competitive module $\bar{g}_{RE}^{(1,3)}(x)$ and the cooperative module $\bar{g}_{M}^{(1,2)}(x)$ are activated. When not controlled by a module, the inputs are applied through user defined maintain velocity commands. The left column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,2)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The right column of the top figure depicts 4 snap shots of the flow $\phi(t, x, \mathbf{u})$ projected onto the space of constraint $X_1^{(1,3)}$, with the red dot corresponding to the snapshot time and blue dots corresponding to recent history. The red sets depict the bad sets $\mathbf{B}_M^{(1,2)}$ and $\mathbf{B}_{RE}^{(1,3)}$ as projected onto the relevant spaces of constraint $X_1^{(1,2)}$ and $X_1^{1,3}$. In the left column, for the current speed, the yellow set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{H}})$ and the green set represents the slice of the restricted capture set $C_{RE}^{(1,3)}(\mathbf{u}_{\mathcal{L}})$. In the right column, the lower boundary for the slice of the capture set $C_{RE}^{(1,3)}$ corresponding to the current speed is shown as a black line. Plots of the control evaluation, inputs applied, and velocities for each vehicle are depicted in the lower plots.

CHAPTER IV

Cooperative Collision Avoidance on Full-Size Vehicles

In this Chapter, we provide computationally efficient decentralized algorithms for a two-vehicle cooperative collision avoidance at traffic intersections, which leverage vehicle-to-vehicle (V2V) communication. Our algorithms are guaranteed to maintain safety (no collision) by design and are not conservative, that is, the driver is overridden by the on-board controller only when it is absolutely needed to prevent a crash. Additionally, the proposed algorithms are designed to be robust to communication delays, sensor noise, and model uncertainty in the vehicle dynamics. We implement our algorithms on two instrumented full scale vehicles and demonstrate their performance on a number of experiments at a traffic intersection. In no instance of the experiments safety was violated. We illustrate also how one can tradeoff between conservatism and aggressiveness by tuning suitable design parameters.

4.1 Organization

This chapter is organized as follows. In Section 4.2, we provide an overview of the problem considered and in Section 4.3 we illustrate our general approach to the problem solution. In Section 4.4, we analyze the vehicle dynamics in detail and illustrate how that fits the class of systems described in Section 4.3. System identification of the vehicle dynam-

ics is also performed in this section. In Section 4.5, we illustrate the software components of the ICA application, including estimation, communication, and control. In Section 4.6, we illustrate the experimental results.

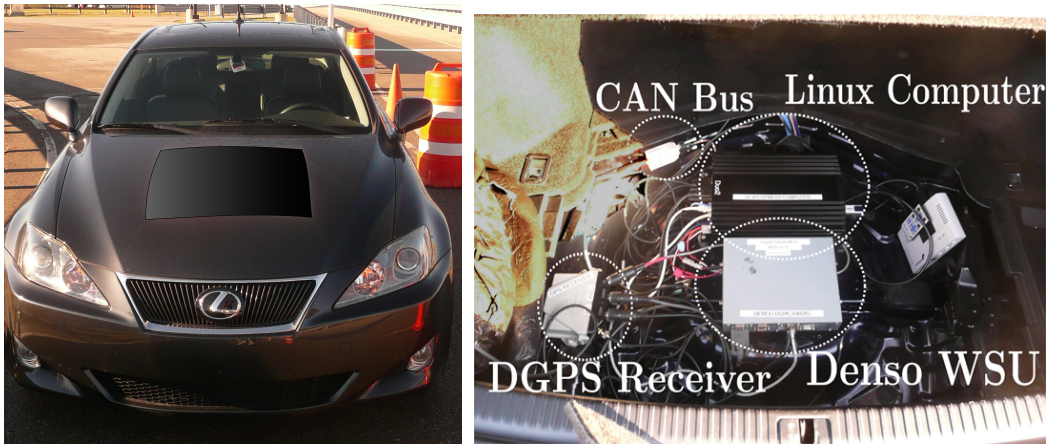
4.2 Problem Overview

We consider the intersection scenario in which two vehicles approach an intersection and can potentially collide in the indicated red shaded area. A collision may occur for a number of reasons, including a distracted driver not seeing the incoming vehicle, under-estimating the vehicle speed, and violating red lights or stop signs. We seek to design controllers on board of each vehicle that use V2V communication in order to negotiate the intersection and apply automatic control only when it is absolutely necessary to prevent a collision.

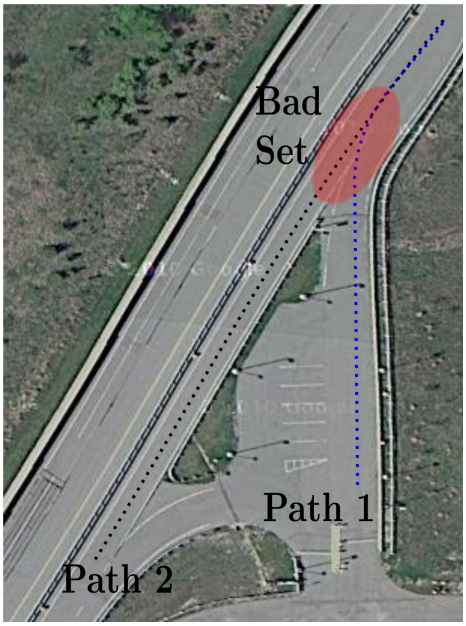
We assume that, after making high level route decisions, drivers follow predefined paths as established by driving lanes. Collisions between two vehicles are prevented by only controlling the longitudinal velocity and displacement of each vehicle along its path, never controlling vehicle steering. We assume each vehicle is equipped with sensors for state measurement (absolute position, heading, velocity, acceleration, brake torque, and pedal position), V2V communication, and the ability to automatically actuate the throttle and brake. Under the above assumptions, the safety algorithms that we illustrate here guarantee that the vehicles will never collide.

4.2.1 Test vehicles and test track

The test vehicles used in this work are modified Lexus IS 250 (2007) test vehicles (Figure 4.1(a)). The modifications include: computer running a Linux operating system; Differential Global Positioning System (DGPS) for position, absolute time and heading measurement; Denso Wireless Safety Unit (WSU) capable of V2V and Vehicle-to-Infrastructure (V2I) Dedicated Short-Range Communications (DSRC); connection to the Controller-



(a) (b)



(c)

Figure 4.1: (a) Modified Lexus IS 250 vehicles used in the experiments. (b) System components are highlighted: main computer running the application, DGPS receiver, computer interface with the CAN bus, and Denso WSU. (c) Top-down view of the test-track

Area Network (CAN) bus to read information from vehicle sensors (velocity, acceleration, brake pedal position, transmission state, etc.); (e) CAN bus interface with brake and throttle actuators. We provide illustrations of some of these systems in Figure 4.1(b).

The computer system is affixed inside the wheel well (Figure 4.1(b)). The purpose of this system is to interface with all on-board vehicle sensors and actuators, in a manner that allows for rapid development, deployment and testing of software applications. The

computer runs an Ubuntu Linux distribution, and consists of a Intel Core-Duo 2.0 GHz processor, 1 GB RAM, 150 GB hard drive, and a motherboard with on-board ethernet and USB ports. A USB video card is connected to the vehicle navigation display unit, and a wireless keyboard is used to control the computer from the passenger seat. The computer can read and write to the CAN bus via a USB adapter. To communicate between vehicles and interface with a DGPS unit, a Denso Wireless Safety Unit (WSU) is connected via ethernet, which is an after-market industry standard (planned) in communication and control for vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) safety systems [76].

The on-board DGPS unit is capable of 0.45 m accuracy for absolute position, 1.5^o accuracy for absolute heading, and 0.1 s accuracy for absolute time. The measurement update rate is 10 Hz. Other sensors include: (i) accelerometer, based on MEMS technology, capable of 0.5 m/s accuracy; (ii) speedometer, measuring average speed at the wheel, capable of 0.5 m/s accuracy; (iii) throttle pedal measurement, capable of 0.5 % accuracy; (iv) brake torque applied at wheel, capable of 0.5 Nm accuracy. The vehicle brake controller is modified to accept brake commands from the computer via CAN bus messages. The drive-by-wire (sends ECU electric signals over CAN bus) throttle pedal, is modified to allow computer issued commands via CAN bus messages to create throttle pedal signals to the ECU. Communication is carried out by the Denso WSU unit. The message standard is the Dedicated Short-Range Communication (DSRC), which is broadcast at the 5.9 GHz band, which is dedicated to V2V and V2I communication. The WSU is connected to a top mounted antenna (Figure 4.1(a)). Communication is carried out with a broadcast network topology, that is, messages transmitted by a sender can be received by any listener in-range.

4.3 Solution Approach

The general solution approach is based on formally encoding the requirement of no-collision into a bad set of vehicle speed and position configurations to be avoided. Then, based on the vehicles dynamical model, we calculate the capture set, which is the set of all vehicle configurations that enter the bad set independently of any throttle/brake control action. Once the capture set is computed, we determine a throttle/brake control map for both vehicles that keeps the system state outside of the capture set at all times. This control map applies throttle and brake inputs only when the system configuration hits the boundary of the capture set. Otherwise, no control action is applied and the driver has full control of the vehicle.

The computation of the capture set and of the control map are usually very demanding, require an exact description of the system dynamics, and assume perfect information on the state of the system. In this section, we illustrate the approach to compute the capture set and the control map developed in [50], which exploits the specific structure of the application domain to overcome these limitations. Specifically it provides efficient algorithms, allows a coarser model obtained from suitable experiments, and is robust to imperfect state information due to sensor uncertainty and especially to communication delays.

4.3.1 System model and safety specification

We model each vehicle as a system Σ^i for $i \in \{1, 2\}$, describing the longitudinal dynamics of vehicle i along its path. Each system Σ^i is an input-output system, defined by the tuple $\Sigma^i := \{X^i, O^i, \mathcal{U}^i, \mathcal{D}^i, f^i, h^i\}$, where $X^i \subset \mathbb{R}^2$ is the state space describing position and speed, $O^i \subset \mathbb{R}^m$ is the output measurement space, $\mathcal{U}^i := [u_L^i, u_H^i] \subset [0, 1] \times [0, 1]$ is the control input space representing the percentage the brake and throttle pedal are depressed, $\mathcal{D}^i := [d_L^i, d_H^i] \subset \mathbb{R}^m$ is the disturbance input space, which can be employed to account for

unmodeled dynamics, $f^i : X^i \times \mathcal{U}^i \times \mathcal{D}^i \to X^i$ is the vector field modeling the dynamics of the vehicle, and $h^i : O^i \rightrightarrows X^i$ is the output set-valued map that provides the set of states compatible with an output measurement. We let $x_1^i \in X_1^i$ denote the longitudinal displacement of vehicle i along its fixed path and x_2^i denote the longitudinal speed of vehicle i along its path. We denote the continuous flow of system Σ^i as $\phi^i(t, x^i, \mathbf{u}^i, \mathbf{d}^i)$, where t denotes the time, x^i denotes the initial state, $\mathbf{u}^i$ denotes the control input signal and $\mathbf{d}^i$ denotes the disturbance signal. In this paper, we will denote in bold signals, which are functions of time.

The two-vehicle system is modeled as the parallel composition of the two systems, denoted as $\Sigma = \Sigma^1 \| \Sigma^2 = \{X, O, \mathcal{U}, \mathcal{D}, f, h\}$, in which $X = X^1 \times X^2$, $O = O^1 \times O^2$, $\mathcal{U} = \mathcal{U}^1 \times \mathcal{U}^2$, $\mathcal{D} = \mathcal{D}^1 \times \mathcal{D}^2$, $f = (f^1, f^2)$, and $h = (h^1, h^2)$. Accordingly, we will let $x = (x^1, x^2)$, $u = (u^1, u^2)$, and $d = (d^1, d^2)$. Furthermore, we let $x_1 = (x_1^1, x_1^2) \in X_1$ denote the pair of two-vehicle displacements. The safety specification for Σ is described in terms of a subset of the state space that needs to be avoided to prevent a collision. Specifically, we call such a set the *bad set* $\mathbf{B} \subset X$ and we will say that the system is safe if the flow never enters the bad set $\mathbf{B}$. For some initial state x_o, the system is safe if there exists some control input signal $\mathbf{u}$ such that for all disturbance input signals $\mathbf{d}$ and time t, we have that $\phi(t, x_o, \mathbf{u}, \mathbf{d}) \notin \mathbf{B}$.

From the construction of the state space and the fact that a collision between two vehicles results when they are both in the red shaded area of Figure 4.2 (a), $\mathbf{B} \subseteq X$ can be defined as

$$\mathbf{B} := \{x \in X \mid (x_1^1, x_1^2) \in]L^1, H^1[\times]L^2, H^2[\}, \tag{4.1}$$

where $L^i < H^i$ for $i \in \{1, 2\}$ (see Figure 4.2 (b)). We also denote $L = (L^1, L^2)$ and $H = (H^1, H^2)$.

The safe controller is based on computing a subset of the state space, called the *capture* set, denoted $C \subseteq X$. The capture set is the set of all initial conditions, such that no control

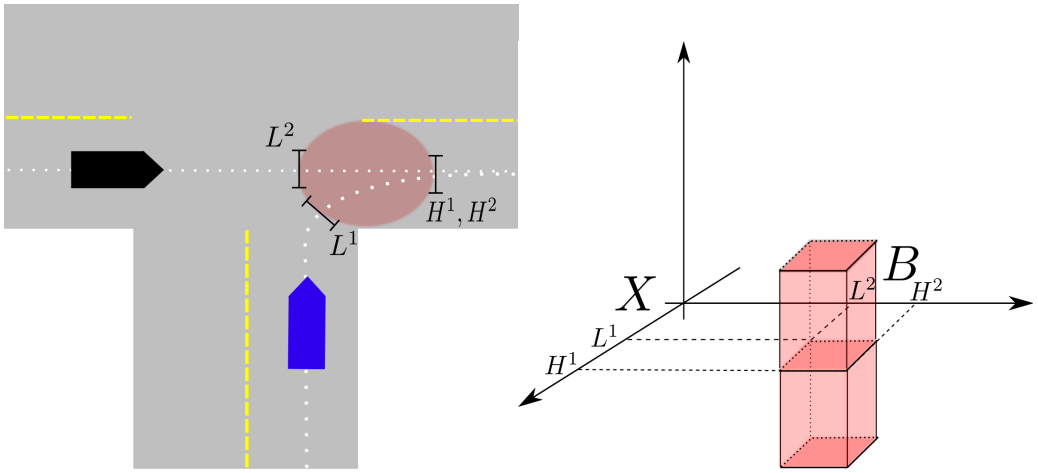


Figure 4.2: Shape of the bad set **B**. (a) L^i determines the lower limit of the collision set along vehicle i path, while U^i determines the upper limit of the collision set along vehicle i path. (b) In the coordinate system where displacement is along the longitudinal path, the bad set **B** is the interval $]L^1, H^1[\times]L^2, H^2[$ in the X_1 space for every value of the speeds of the two vehicles.

input can prevent a collision. The mathematical definition is given by

$$C := \{x \in X \mid \forall\, \mathbf{u},\ \exists\, t,\ \ \exists\, \mathbf{d} \text{ s.t. } \phi(t, x, \mathbf{u}, \mathbf{d}) \in \mathbf{B}\}. \tag{4.2}$$

The approach of our solution to the safety control problem is to compute the capture set, and through the application of feedback control, prevent the flow from ever entering the capture set. By the definition of the capture set, safety is guaranteed if the flow never enters the capture set.

Computing the capture set is in general a difficult problem. In the next sections, we show how exploiting the structural features of the specific system under study allows us to compute this set and handle imperfect state information.

4.3.2 Computation approach exploiting partial orders

In this section, we illustrate the main result of [50] to compute the capture set. This approach relies on (i) the state and input spaces of the system Σ^i being partially ordered and (ii) the flow of the system Σ^i being an order preserving map. Specifically, for the state space $X^i \subseteq \mathbb{R}^2$, we consider elements to be partially ordered according to component-wise

ordering , that is, for $z^i, w^i \in X^i$ we have that $z^i \leq w^i$ provided $z^i_1 \leq w^i_1$ and $z^i_2 \leq w^i_2$. Further, we consider the partial ordering between input signals defined for signals $\mathbf{u}^i, \mathbf{v}^i$ as $\mathbf{u}^i \leq \mathbf{v}^i \Leftrightarrow \mathbf{u}^i(t) \leq \mathbf{v}^i(t)$ for all t. The inequality $\mathbf{u}^i(t) \leq \mathbf{v}^i(t)$ is defined such that $\mathbf{u}^i_1(t) \geq \mathbf{v}^i_1(t)$ and $\mathbf{u}^i_2(t) \leq \mathbf{v}^i_2(t)$. We assume that the flow of each system Σ^i is an *order preserving* map. Mathematically, this means that for initial conditions $z^i, w^i \in X^i$, inputs $\mathbf{u}^i, \mathbf{v}^i$ and disturbances $\mathbf{d}^i, \mathbf{b}^i$, the following implication holds

$$z^i \leq w^i \ \wedge \ \mathbf{u}^i \leq \mathbf{v}^i \ \wedge \ \mathbf{d}^i \leq \mathbf{b}^i \Rightarrow \phi^i(t, z^i, \mathbf{u}^i, \mathbf{d}^i) \leq \phi^i(t, w^i, \mathbf{v}^i, \mathbf{b}^i) \ \forall \ t. \tag{4.3}$$

In terms of the vehicle dynamics, this assumption implies that greater initial displacement, greater initial velocity, and greater inputs will lead to greater displacements and speeds at any time. The validity of this assumption for the vehicle dynamics is discussed in detail in Section 4.4, where the vehicle model is introduced. A liveliness condition is introduced by requiring that for at least one i $f^i_1(x^i, u^i, d^i) > 0$ for all x^i, u^i and d^i. From a practical point of view, this requires that vehicle i does not go in reverse and does not stop.

The order preserving property of the dynamics along with the structure of the bad set can be exploited to compute the capture set for system $\Sigma = \Sigma^1 || \Sigma^2$ with an algorithm that has linear complexity with respect to the state dimension. The algorithm is based on the *restricted* capture set, which for a fixed input signal $\mathbf{u}$, is defined as $C_{\mathbf{u}} := \{x \in X \mid \exists \ t \geq 0, \ \exists \ \mathbf{d} \ \text{ s.t. } \phi(t, x, \mathbf{u}, \mathbf{d}) \in \mathbf{B}\}$. This set represents the set of initial conditions that are taken into the bad set under the *fixed* input signal $\mathbf{u}$. Define the fixed input signals $\mathbf{u}_{\mathcal{L}}, \mathbf{u}_{\mathcal{H}}$, as $\mathbf{u}_{\mathcal{L}}(t) := (u^1_H, u^2_L)$ and $\mathbf{u}_{\mathcal{H}}(t) := (u^1_L, u^2_H)$ for all t. Then, we have ([50])

$$C = C_{\mathbf{u}_{\mathcal{L}}} \cap C_{\mathbf{u}_{\mathcal{H}}}. \tag{4.4}$$

The capture set can be computed by only computing the two restricted capture sets corresponding to maximum and minimum inputs. The restricted capture sets are simpler to compute, since they can be obtained by just integrating the dynamics under fixed control

inputs. This is in contrast with the capture set C, whose computation requires the solution of a differential game between the control and the disturbance.

Based on the expression of the capture set given in (4.4), the feedback control map is given by

$$g(x) := \begin{cases} (u_H^1, u_L^2) & \text{if } x \in C_{\mathbf{u}_\mathcal{L}} \text{ and } x \in \partial C_{\mathbf{u}_\mathcal{H}}, \\ (u_L^1, u_H^2) & \text{if } x \in \partial C_{\mathbf{u}_\mathcal{L}} \text{ and } x \in \overline{C_{\mathbf{u}_\mathcal{H}}}, \\ \mathcal{U} & \text{otherwise,} \end{cases} \tag{4.5}$$

in which $\overline{C_{\mathbf{u}_\mathcal{H}}}$ denotes the closure of $C_{\mathbf{u}_\mathcal{H}}$. The controller allows the driver to chose any input until the flow hits the boundary of the capture set. The driver retains control once the flow no longer touches the boundary of the capture set. A visual interpretation of the feedback map is provided in Figure 4.3.

In the presence of communication delays and/or uncertain sensor readings the vehicles will not have access to the exact value of the system state but to a set of possible current system states. This can be easily incorporated in the above described control strategy [50]. Let the set of possible current system states be denoted $\hat{x} \subset X$, which can be constructed using output measurement $z \in O$ as explained in Section 4.5.1. The safety specification is now posed in terms of preventing the state uncertainty $\hat{x}$ from intersecting the bad set $\mathbf{B}$. That is, the system is safe if $\hat{x}(t) \cap \mathbf{B} = \emptyset$ for all $t \in \mathbb{R}_+$. It has been shown that this is the case if and only if $\hat{x}(t)$ never intersects both $C_{\mathbf{u}_\mathcal{L}}$ and $C_{\mathbf{u}_\mathcal{H}}$ at the same time [50]. The feedback set-valued map g, as defined in (4.5) can still guarantee this as long as it is extended to set $\hat{x}$ as follows

$$g(\hat{x}) := \begin{cases} (u_H^1, u_L^2) & \text{if } \hat{x} \cap C_{\mathbf{u}_\mathcal{H}} \neq \emptyset \text{ and } \hat{x} \cap \partial C_{\mathbf{u}_\mathcal{L}} \neq \emptyset \text{ and } \hat{x} \cap C_{\mathbf{u}_\mathcal{L}} = \emptyset, \\ (u_L^1, u_H^2) & \text{if } \hat{x} \cap \overline{C_{\mathbf{u}_\mathcal{L}}} \neq \emptyset \text{ and } \hat{x} \cap \partial C_{\mathbf{u}_\mathcal{H}} \neq \emptyset \text{ and } \hat{x} \cap C_{\mathbf{u}_\mathcal{H}} = \emptyset, \\ \mathcal{U} & \text{otherwise.} \end{cases} \tag{4.6}$$

The interpretation of this feedback set-valued map is that control is applied when the

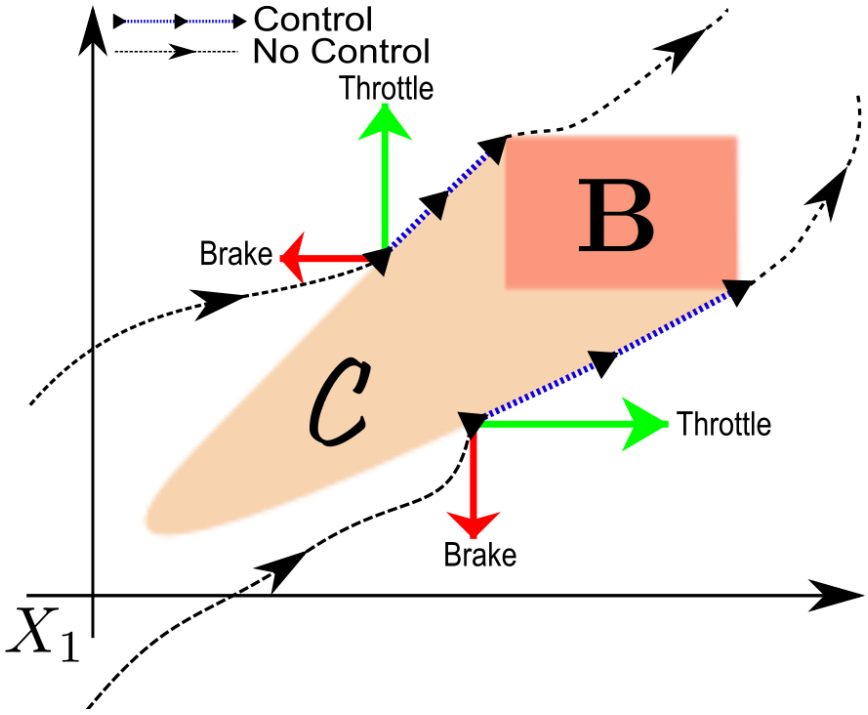


Figure 4.3: Feedback map $g(x)$ shown for two separate trajectories. The pink region represents a slice of the capture set in position space corresponding to a pair of vehicles speeds. When the flow touches the upper boundary of the capture set, geometrically as $x \in C_{\mathbf{u}_{\mathcal{L}}}$ and $x \in \partial C_{\mathbf{u}_{\mathcal{H}}}$, the feedback controller commands the input (u_L^1, u_H^2), corresponding to vehicle 1 applying maximum brake while vehicle 2 applies maximum throttle. When the flow touches the lower boundary of the capture set, geometrically as $x \in \overline{C_{\mathbf{u}_{\mathcal{H}}}}$ and $x \in \partial C_{\mathbf{u}_{\mathcal{L}}}$, the feedback controller commands the input (u_H^1, u_L^2), corresponding to vehicle 1 applying maximum throttle while vehicle 2 applies maximum brake.

state uncertainty has non-empty intersection with either $C_{\mathbf{u}_{\mathcal{L}}}$ or $C_{\mathbf{u}_{\mathcal{H}}}$, and simultaneously is touching the boundary of the other. We remark that by construction, feedback map g is order reversing with respect to partial order established by set inclusion, that is, $A \subset B \implies g(A) \supset g(B)$. This property implies that the larger the state uncertainty, the more conservative the controller will be.

4.3.3 Algorithmic Implementation

In this section, we provide a summary of the algorithms that compute the restricted capture set for the case in which the first component of the vector fields f^i do not depend on the x_1^i coordinate (displacement) [50]. This assumption is satisfied by the vehicle dynam-

ics considered in the next section. The algorithms are implemented on-board the vehicle computer, therefore they must use a discrete-time model of the dynamics. For $n > 0$ and step size $\Delta T > 0$, the discrete-time flow of system Σ is given by $\Phi(n, x, \mathbf{u}, \mathbf{d})$ and is generated by the forward Euler approximation of the continuous time dynamics, mathematically given by $\Phi(n+1, x, \mathbf{u}, \mathbf{d}) = \Phi(n, x, \mathbf{u}, \mathbf{d}) + \Delta T\, f(\Phi(n, x, \mathbf{u}, \mathbf{d}), \mathbf{u}[n-1], \mathbf{d}[n-1])$, with initial condition $\Phi(0, x, \mathbf{u}, \mathbf{d}) = x$, and sampled signals $\mathbf{u}[n] := \mathbf{u}(n\Delta T)$ and $\mathbf{d}[n] := \mathbf{d}(n\Delta T)$.

The feedback map g is implemented in discrete time, which requires an alternate definition of the capture set boundary. We will say that the set $\hat{x}[n] \subset X$ intersects the boundary and not the interior of the restricted capture set $C_{\mathbf{u}}$ provided $\hat{x}[n] \cap C_{\mathbf{u}} = \emptyset$ and $\hat{x}[n+1] \cap C_{\mathbf{u}} \neq \emptyset$. This states that $\hat{x}[n]$ intersects the boundary and not the interior of the restricted capture set if it is currently outside of the set, but it will be inside the set at the next time step.

To compute the capture set $C_{\mathbf{u}}$, we can compute a *slice* of it in the displacement space, denoted $\mathscr{C}_{\mathbf{u}} \subset X_1$, corresponding to the current two-vehicle velocity (x_2^1, x_2^2). Due to the order preserving properties of the dynamics with respect to state and input, and the structure of the bad set $\mathbf{B}$, the restricted capture set slice is computed through the back propagation of the upper and lower bounds of the bad set, i.e., $L, H \in X_1$. Specifically, define the sequences

$$\begin{aligned} L(n, x, u) &:= L + x_1 - \Phi_1(n, x, \mathbf{u}, \mathbf{d}_H), \\ H(n, x, u) &:= H + x_1 - \Phi_1(n, x, \mathbf{u}, \mathbf{d}_L), \end{aligned} \tag{4.7}$$

where $\mathbf{d}_L(k) := (d_L^1, d_L^2)$ and $\mathbf{d}_H(k) := (d_H^1, d_H^2)$ for all k. Given current state estimate set $\hat{x}$, the restricted capture set slice $\mathscr{C}_{\mathbf{u}}$ can be written as (Algorithm 1)

$$\mathscr{C}_{\mathbf{u}} = \bigcup_{k \in \mathbb{N}}]L(n, \sup \hat{x}, \mathbf{u}), H(n, \inf \hat{x}, \mathbf{u})[.$$

Membership within the capture set slice can then be concluded by taking intersection of

the state uncertainty with the collection of all interval sets, established by

$$\hat{x}_1 \cap \bigcup_{k\in\mathbb{N}}]L(n, \sup \hat{x}, \mathbf{u}), H(n, \inf \hat{x}, \mathbf{u})[\neq \emptyset \Leftrightarrow \hat{x}_1 \cap \mathscr{C}_{\mathbf{u}} \neq \emptyset. \tag{4.8}$$

Algorithm 5 $\mathscr{C}_{\mathbf{u}}$ = CaptureSetSlice($\hat{x}, \mathbf{u}$)

Input: $(\hat{x}, \mathbf{u}) \in 2^X \times S(\mathcal{U})$

$n = 1$
loop
  if $\inf \hat{x}_1 \leq H(n, \inf \hat{x}, \mathbf{u})$ and $\inf \hat{x}_1 \notin]L(n, \sup \hat{x}, \mathbf{u}), H(n, \inf \hat{x}, \mathbf{u})[$ **then**
    $n = n + 1$
  else
    return $\mathscr{C}_{\mathbf{u}} = \bigcup_{k\leq n}]L(k, \sup \hat{x}, \mathbf{u}), H(k, \inf \hat{x}, \mathbf{u})[$.
  end if
end loop

Output: $\mathscr{C}_{\mathbf{u}} \subset X_1$.

We can determine non-empty intersection of the capture set with the state uncertainty by using the equivalence $\hat{x}_1 \cap \mathscr{C}_{\mathbf{u}} = \emptyset \Leftrightarrow \hat{x} \cap C_{\mathbf{u}} = \emptyset$. The closed-loop implementation of the feedback map (4.6), in discrete time, is provided in Algorithm 2, where $\mathbf{u}$ = FeedbackMap($\hat{x}[n+1], \hat{x}[n]$).

Note that for evaluating the control map, we only need to calculate the sequences $L(n, x, u)$ and $H(n, x, u)$ for two extremal constant inputs $u = (u_H^1, u_L^2)$ and $u = (u_L^1, u_H^2)$. Hence, we do not require the detailed model of the system Σ, we just need to know how the system responds to these two extremal constant inputs. As we will see in Section 4.4, this can be achieved through a series of experiments where these constant inputs are applied for a set of different initial speeds.

4.4 Vehicle Dynamics

The vehicle dynamics, which takes throttle and brake as inputs and provides longitudinal displacement as output, is the cascade of the powertrain system and the vehicle model (Figure 4.4(a)). The powertrain system (Figure 4.4(b)) generates the wheel torque inputs in

Algorithm 6 $u = \text{FeedbackMap}(\hat{x}[n+1], \hat{x}[n])$

Input: $(\hat{x}[n+1], \hat{x}[n]) \in 2^X \times 2^X$

Construct capture set slices for state prediction.
$\mathscr{C}_{\mathbf{u}_\mathcal{L}} = \text{CaptureSetSlice}(\hat{x}[n+1], \mathbf{u}_\mathcal{L})$, $\mathscr{C}_{\mathbf{u}_\mathcal{H}} = \text{CaptureSetSlice}(\hat{x}[n+1], \mathbf{u}_\mathcal{H})$

Check if predicted state $\hat{x}[n+1]$ intersects both capture set slices.
if $\hat{x}[n+1] \cap \mathscr{C}_{\mathbf{u}_\mathcal{L}} \neq \emptyset$ *and* $\hat{x}[n+1] \cap \mathscr{C}_{\mathbf{u}_\mathcal{H}} \neq \emptyset$ **then**

Construct capture set slices for current state.
$\mathscr{C}_{\mathbf{u}_\mathcal{L}} = \text{CaptureSetSlice}(\hat{x}[n], \mathbf{u}_\mathcal{L})$, $\mathscr{C}_{\mathbf{u}_\mathcal{H}} = \text{CaptureSetSlice}(\hat{x}[n], \mathbf{u}_\mathcal{H})$

Determine control according to equation (4.6).
if $\hat{x}_1[n] \cap \mathscr{C}_{\mathbf{u}_\mathcal{L}} = \emptyset$ *and* $\hat{x}_1[n] \cap \mathscr{C}_{\mathbf{u}_\mathcal{H}} \neq \emptyset$ **then**
$u = u_\mathcal{L}$
else if $\hat{x}_1[n] \cap \mathscr{C}_{\mathbf{u}_\mathcal{L}} \neq \emptyset$ *and* $\hat{x}_1[n] \cap \mathscr{C}_{\mathbf{u}_\mathcal{H}} = \emptyset$ **then**
$u = u_\mathcal{H}$
else
$u = u_\mathcal{L}$
end if

else
No control specified.
$u \in \mathcal{U}$
end if

Output: $u \in \mathcal{U}$.

response to throttle and brake inputs. The vehicle model takes throttle and brake inputs and produces longitudinal displacement as output according to Newton's law. In this section, we describe each of the two subsystems and illustrate how the cascade of the two generates a flow that is an order preserving map when throttle inputs do not change with time. Then, we perform a system identification procedure to determine the dynamics of the cascade system only in response to maximal throttle and maximal braking, which is sufficient for the implementation of the control map as described in Section 4.3.

4.4.1 Vehicle Model

The longitudinal displacement of the vehicle along its path is denoted by p and the longitudinal velocity is denoted by $v \in [v_{min}, v_{max}]$, where $v_{min} \geq 0$. The controlled forces that act on the vehicle are the brake input $f_b \in \mathcal{F}_b = [f_{min}, 0]$ with $f_{min} < 0$ and engine input

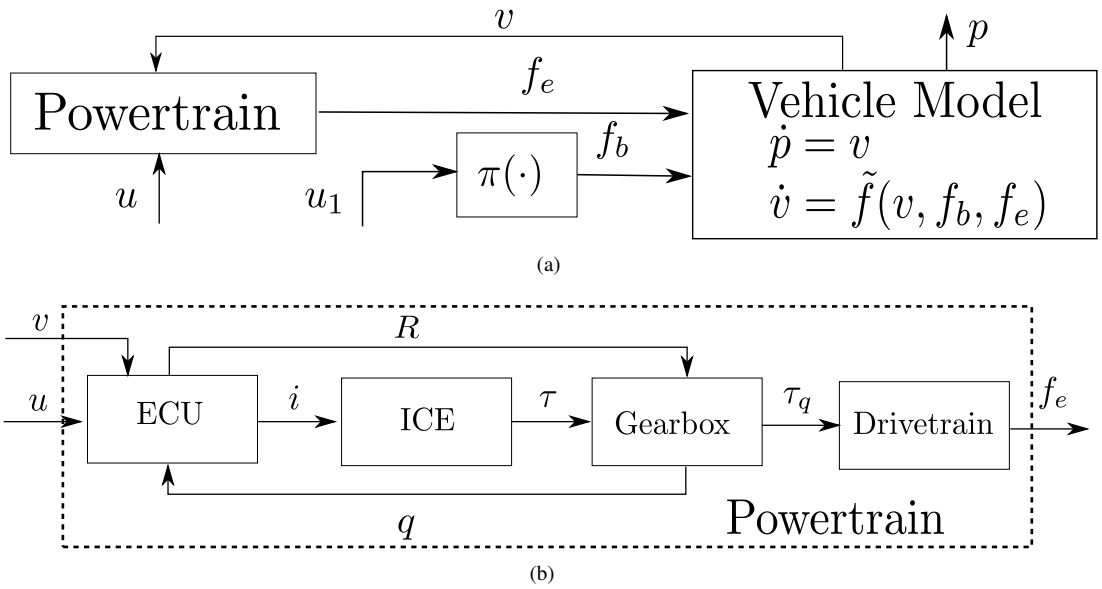


Figure 4.4: (a) Block diagram representing the cascade of the powertrain model and the vehicle model. Here, p denotes longitudinal displacement and v denotes longitudinal speed. The powertrain model (b) takes the inputs u and velocity v to produce engine torque at the wheel f_e. The static map π takes the brake pedal percentage input u_1 to produce brake torque f_b. The vehicle model takes the brake force f_b and engine force f_e as inputs. (b) Powertrain system. The Engine Control Unit (ECU) is a means of controlling the fuel injection rate and the gear state q of the transmission. The output signals of the ECU are the fuel injection rate i and the gear reset R. The second block is the Internal Combustion Engine (ICE), which is where the fuel combustion takes place based on the fuel injection rate i, and produces an output torque τ at the flywheel. The next block is the transmission, which converts torque at the flywheel τ to torque at the transmission output τ_q as a function of the gear state q. The drivetrain is the last block, which transfers torque from the gearbox τ_q to force at the wheel f_e.

$f_e \in \mathcal{F}_e = [0, f_{max}]$ with $f_{max} > 0$. The brake force f_b is controlled by the driver via the surjective-monotone map $\pi : \mathcal{U}_1 \to \mathcal{F}_b$ that takes brake pedal percentage u_1 as an input, while the engine force f_e is supplied by the powertrain (Figure 4.4(a)). The longitudinal dynamics are given by

$$\frac{dv}{dt} = \frac{\mathcal{R}^2}{J_w + \mathcal{M}\mathcal{R}^2}(f_e + f_b - \frac{\rho_{air}}{2} C_D A_f v^2 - C_{rr}\mathcal{M}g) =: \tilde{f}(v, f_b, f_e), \tag{4.9}$$

where $\mathcal{R}$ is the wheel radius, $\mathcal{M}$ is the vehicle mass, ρ_{air} is the air density, C_D is the air drag coefficient, A_f is the projected vehicle cross section, and C_{rr} is the coefficient of rolling friction [98].

The longitudinal dynamics (4.9) generate a flow $(p(t, p_o, v_o, \mathbf{f_b}, \mathbf{f_e}), v(t, v_o, \mathbf{f_b}, \mathbf{f_e}))$ that is

an order preserving map with respect to brake force input signal $\mathbf{f}_b$, engine force signal $\mathbf{f}_e$, and initial conditions (p_o, v_o). That is, larger forces f_b and f_e will result in greater displacements and speeds; larger initial conditions (p_o, v_o) will also result in larger displacements and speeds. On the input space, we use the partial order defined by by $u \leq v$ provided $u_1 \geq v_1$ and $u_2 \leq v_2$. Consequently, we have $u_L = (1, 0)$ and $u_H = (0, 1)$. Since the brake force map $\pi : \mathcal{U}_1 \to \mathcal{F}_b$ is monotone, the flow is an order preserving map also with respect to the brake input $\mathbf{u}_1$. In the next section, we illustrate the components of the powertrain.

4.4.2 Powertrain

The dynamics of the powertrain take as control inputs $u = (u_1, u_2) \in [0, 1] \times [0, 1]$, where the first component u_1 denotes the brake pedal percent input, and the second component u_2 denotes the throttle pedal percent input [25]. In our application, these inputs can be administered either by the driver or by the automatic controller. The output of the system is assumed to be the torque applied at the wheel of the vehicle f_e. An overview of the system is provided in Figure 4.4(b).

The first component of the powertrain is the Engine Control Unit (ECU). This subsystem determines the fuel injection rate $i \in [0, 1]$ into the Internal Combustion Engine (ICE), and the current gear $q \in \{1, 2, 3, 4, 5, 6\}$ of the gearbox. The inputs to this block consist of the current velocity of the vehicle v, the throttle pedal input u_2 and the brake pedal input u_1. The second component of the powertrain is the Internal Combustion Engine (ICE). The output of this system is the torque τ applied by the flywheel, and the input is the fuel injection rate administered by the ECU. The third component of the powertrain is the gearbox. This module consists of the transmission with a fixed gear ratio. All switching logic is determined by the ECU, which sends a reset input R to the gearbox when a gear shift has been determined. The gearbox takes torque at the flywheel τ and converts it

to the torque τ_q based on the current gear. The last component of the powertrain is the drivetrain. This component transfers torque at the gearbox τ_q to force applied at the wheel f_e. This module consists of the flywheel, torque converter, variable gear ratio transformer, propeller shaft, final drive and drive shaft (details can be found, for example, in [97]).

For the powertrain model, the order preserving property of the output f_e with respect to throttle input u_2 does not hold in general. This is due to the complexity of the ECU, which controls the fuel injection rate in a manner that optimizes a set of performance metrics, such as emissions, engine thermodynamic efficiency, with transients that can be quite complex and non-monotone [25]. By design, however, this is performed in a manner that generates monotone input-output behavior at *steady-state* [42].

Therefore, the dynamics of the vehicle system that take brake u_1 and throttle u_2 commands as inputs and provide speed and displacement as output are order preserving with respect to constant throttle input at least after an initial transient. Hence, we restrict the control commands to be constant with time, so that the system dynamics generate an order preserving flow with respect to the inputs after an initial transient time ϵ. In the next section, we illustrate how to identify the vehicle dynamics for the maximal braking and throttle inputs, which is the only knowledge on the model required by our algorithm.

4.4.3 System Identification

In order to model how the powertrain responds to constant control inputs (maximal braking and maximal throttle), in principle one should model the details of all the blocks in Figure 4.4(b). Rather than modeling this level of detail, we exploit the fact that the approach illustrated in Section 4.3 allows for disturbance inputs, which we use here to account for unmodeled dynamics. For the input signal $\mathbf{u}$ and velocity signal $\mathbf{v}$, define the non-deterministic engine force trajectories $\mathbf{F}_e(\mathbf{u}, \mathbf{v})$ as the set of all possible output engine

force trajectories applied at the wheel given an input signal and velocity signal. When the powertrain model is combined with the vehicle physics, the vehicle velocity v and engine force at the wheel f_e are coupled through the longitudinal dynamics introduced in (4.9). To capture this dependency, we say a system evolution is *realizable* if the velocity trajectory $\mathbf{v}(t, v_0, \mathbf{u}_1, \mathbf{f}_e)$ and engine torque trajectory $\mathbf{f}_e([0,t])$ satisfy (4.9) at all time and the inclusion

$$\mathbf{f}_e([0,t]) \in \mathbf{F}_e(\mathbf{u}([0,t]), \mathbf{v}([0,t], v_0, \pi(\mathbf{u}_1), \mathbf{f}_e)). \tag{4.10}$$

Let $\epsilon \in \mathbb{R}_+$ denote the maximum delay between initial changes in driver input u and steady state vehicle acceleration $\dot{v}$. This is the consequence of delays in: (1) software subsystems of the drive-by-wire throttle system; (2) delays in the powertrain due to chemical combustion; (3) gear shift delays; and (4) delays imposed by the Engine Control Unit (ECU) for filtering and environmental reasons. For a speed x_2, input u^*, and time-delay constant $\epsilon \geq 0$, the *permissible acceleration* set, denoted $\Upsilon(x_2, u^*, \epsilon) \subset \mathbb{R}$, is given by

$$\Upsilon(x_2, u^*, \epsilon) := \left\{ \begin{array}{c} \tilde{f}(\mathbf{v}(t, v_0, \pi(\mathbf{u}_1^*), \mathbf{f}_e), \pi(\mathbf{u}_1^*(t)), \mathbf{f}_e(t)) \in \mathbb{R} \mid \\ \exists\, \mathbf{f}_e([0,t]) \in \mathbf{F}_e(\mathbf{u}^*, \mathbf{v}([0,t], v_0, \pi(\mathbf{u}_1^*), \mathbf{f}_e)), \\ \exists\, t \geq \epsilon,\ \exists\, v_0 \text{ s.t. } x_2 = \mathbf{v}(t, v_0, \pi(\mathbf{u}_1^*), \mathbf{f}_e) \end{array} \right\}, \tag{4.11}$$

where $\mathbf{u}^*(t) = u^*$ for all t. This is the set of all possible accelerations $\tilde{f}(x_2, \pi(u_1^*), \mathbf{f}_e(t))$ achievable at velocity x_2 after $t \geq \epsilon$ seconds have elapsed under the constant input signal $\mathbf{u}^*$. Letting $x_1 = p$ and $x_2 = v$, we construct the vector field $f(x, u, d)$ of Section 4.3.2 for a fixed input $u = u^*$ as follows

$$f_1(x, u^*, d) := x_2, \quad f_2(x, u^*, d_H) := \sup \Upsilon(x_2, u^*, \epsilon), \quad f_2(x, u^*, d_L) := \inf \Upsilon(x_2, u^*, \epsilon).$$

For the case of maximum disturbance d_H (minimum disturbance d_L), the interpretation of $f_2(x, u^*, d_H)$ ($f_2(x, u^*, d_L)$) is that it represents the *greatest* acceleration (*least* acceleration) that can possibly be achieved at the velocity x_2 after the constant input u^* has

been applied for *at least* $\epsilon \geq 0$ seconds. If $\Upsilon(x, u^*, \epsilon) = \emptyset$, then find the minimizer $x_2^* := \arg\min_{y_2 \in X_2} \{\|y_2 - x_2\| \mid \Upsilon(y_2, u^*, \epsilon) \neq \emptyset\}$ and set $f(x, u^*, d) = f((x_1, x_2^*), u^*, d)$.

For implementing the feedback map of Section 4.3.2, it is enough to identify experimentally $f_2(x, u_L, d_H)$ and $f_2(x, u_H, d_L)$. The identification procedure is as follows. To identify $f_2(x, u_L, d_H)$, we conducted a set of experiments called *braking trials*, in which, starting from an initial constant velocity, maximal braking $u_L = (1, 0)$ is applied and vehicle acceleration after $\epsilon = 0.7$s is recorded to provide data points for $\Upsilon(x_2, u_L, \epsilon)$ for the values of speed x_2 reached after ϵ. The value of ϵ was chosen to be enough for the vehicle to reach a steady state acceleration. Several trials for the same initial speed were performed and the infimum of these data points for every speed x_2 was computed to provide the value of $f_2(x, u_L, d_H)$. The set of initial velocities chosen is $\mathcal{V}_0 := \left\{\frac{1}{4}v_{max}, \frac{1}{2}v_{max}, \frac{3}{4}v_{max}, v_{max}\right\}$, in which $v_{max} = 8$ m/s for vehicle 1 (Blue IS 250) and $v_{max} = 17$ m/s for vehicle 2 (Grey IS 250). A brake trial consists of the following steps (1) accelerate each vehicle to a nominal constant velocity $v_0 \in \mathcal{V}_0$ on the vehicle path; (2) maintain velocity v_0 for at least 2 seconds, so transmission comes to a steady state; (3) apply brake input $u_L := (1, 0)$ via computer issued command, driver does not override command until vehicle reaches rest.

Similarly, to identify $f_2(x, u_H, d_L)$, we conducted a set of experiments called *throttle trials*, in which starting from an initial constant velocity, maximal throttle $u_H = (0, 1)$ for the vehicle 1 and $u_H = (0, 0.5)$ for the vehicle 2 was applied. The set of initial velocities are given by $\mathcal{V}_0 := \left\{0, \frac{1}{4}v_{max}, \frac{1}{2}v_{max}, \frac{3}{4}v_{max}\right\}$, in which $v_{max} = 8$ m/s for vehicle 1 and $v_{max} = 17$ m/s for vehicle 2. A throttle trial consists of the following steps: (1) accelerate each vehicle to a nominal constant velocity $v_0 \in \mathcal{V}_0$ on vehicle path, if $v_0 = 0$, leave vehicle in idling state; (2) maintain velocity v_0 for at least 2 seconds, so transmission comes to steady state; (3) apply acceleration input via computer issued command, driver does not override command until vehicle reaches maximum velocity v_{max}.

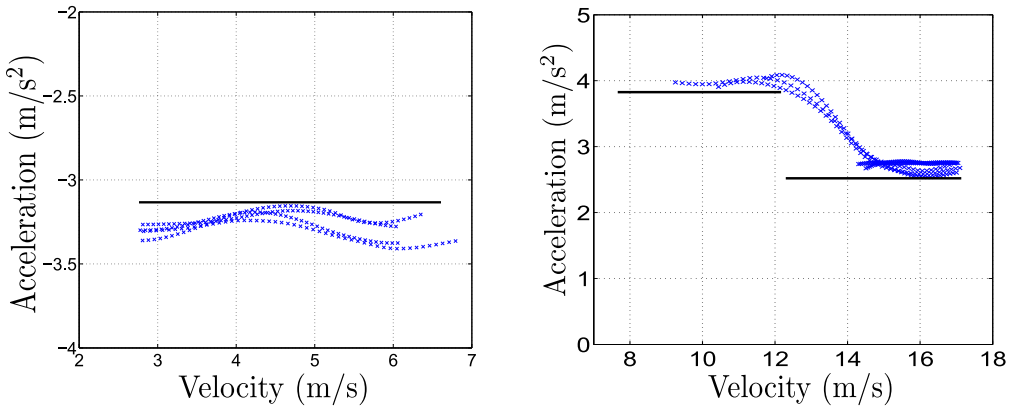
For vehicle 1, which has $\mathcal{U}^1 = [0,1] \times [0,0.5]$ and $x_2^1 \in [0, 8.8]$ m/s, along path 1 (as shown in Figure 4.1(c)), we obtained $f_2^1(x_2^1, u_L^1, d_H^1) = -3.0$ and

$$f_2^1(x_2^1, u_H^1, d_L^1) = \begin{cases} 3.0 & x_2^1 \in [0, 7), \\ 1.75 & x_2^1 \in [7, \infty). \end{cases} \tag{4.12}$$

For vehicle 2, which has $\mathcal{U}^2 = [0,1] \times [0,1]$ and $x_2^2 \in [8.8, 20]$ m/s, along path 2 (as shown in Figure 4.1(c)), we obtained also $f_2^2(x_2^2, u_L^2, d_H^2) = -3.0$ and

$$f_2^2(x_2^2, u_H^2, d_L^2) = \begin{cases} 3.9 & x_2^2 \in [0, 13), \\ 2.5 & x_2^2 \in [13, \infty). \end{cases} \tag{4.13}$$

Figure 4.5 shows the system identification results for vehicle 2. Similar plots were obtained for vehicle 1.



(a) (b)

Figure 4.5: (a) A summary of all the experimental data for identifying $f_2^2(x_2^2, u_L^2, d_H^2)$ (black solid line) of vehicle 2. (b) A summary of all the experimental data for identifying $f_2^2(x_2^2, u_H^2, d_L^2)$ (black solid line) of vehicle 2.

4.5 Software Implementation

The major software components of the ICA application are estimation, communication, and control (Figure 4.6).

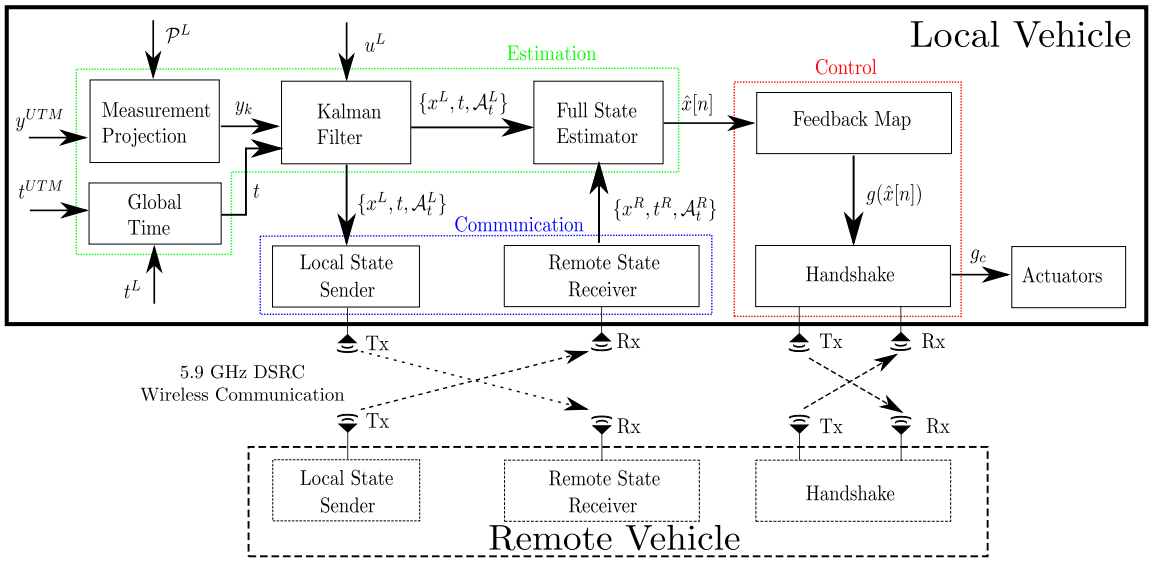


Figure 4.6: Software system overview for the local vehicle. In the figure, we let the superscript L denote the local vehicle while the superscript R denotes the remote vehicle. The estimator (delimited by a green box) takes as inputs the UTM time and position information (y^{UTM} and t^{UTM}), the vehicle path information $\mathcal{P}^L$, the local vehicle time t^L, the local vehicle input u^L, and time/state information of the remote vehicle $\{x^R, t^R, \mathcal{A}_t^R\}$, and provides a set of possible position/speed configurations for the two-vehicle system $\hat{x} \subset X$. The communication system (delimited by the blue box) is a module that continuously sends to and receives information from the remote vehicle. The control system takes as input the state estimate set $\hat{x}$ computed locally and information from the control evaluation from the remote vehicle and returns the control input applied to the vehicle.

4.5.1 Estimation

State estimation consists of several modules: longitudinal state measurement construction from raw measurements in UTM coordinates; calculation of the universal time; Kalman filter for local state prediction; and a full state estimator to construct the current state estimate set $\hat{x}(t) \subset X$ for the whole system. We denote with superscript "L" quantities computed on the local vehicle while with superscript "R" we denote quantities of the remote vehicle that the local vehicle receives through the wireless communication. The measurement projection block is used to compute the longitudinal state measurement y_k from GPS and CAN measurements y^{UTM} (heading and position from GPS, velocity from CAN). The global time is computed by using a local time measurement t^L from the vehicle

PC, and drift is removed by using the universal time t^{UTM} from the GPS system. The Kalman filter combines the longitudinal state measurement y_k and the pedal inputs u^L to compute the state estimate x^L and acceleration profile $\mathcal{A}_t^L$. This information is sent both to the communication system, and to the full state estimator. The full state estimator takes the current state estimate, time and acceleration profile $\{x^L, t^L, \mathcal{A}_t^L\}$, and combines this with the remote state information $\{x^R, t^R, \mathcal{A}_t^R\}$ to construct the full state estimate $\hat{x}[k]$ for use by the controller.

The time measurements available to each vehicle consist of the global time t^{UTM}, taken from the GPS system, and the local time t^L taken off the vehicle PC. The global time t^{UTM} is accurate, however only is received at a rate of 10 Hz, and can sometimes be unavailable due to message loss. The local time t^L is available at a higher rate of 1.5 GHz to a precision of 1 ms, however it is not accurate globally due to inherent drift in the crystal oscillator used to calculate time. To accurately compute a global time with update rate equal to 1.5 GHz, we combine the global time t^{UTM} with the local time t^L to produce the time t with using a simple moving average, where the moving average is updated every time a new global time t^{UTM} is made available.

The measurement projection block constructs a longitudinal state measurement from raw sensors on-board the vehicle. This involves projecting raw measurements onto the vehicle's path stored locally in $\mathcal{P}^L$. The source of absolute position and heading measurements is the GPS system, which provides updates at a fixed broadcast rate of 10Hz.

Kalman filter

For the Kalman filter, the longitudinal dynamics are assumed to be linear and hybrid, where the transmission state $q \in \{1, 2, 3, 4, 5, 6\}$ is assumed to be known at all time as obtained from the CAN bus. To model rolling friction, we add a fictitious frictional input, which takes values based on the sign of velocity, given by $u_3 = \text{sgn}(x_2)$. Since we seek

to estimate also the acceleration, we add the engine torque at the wheels as a third state. Specifically, the Kalman filter state is $\hat{e} \in \mathbb{R}^3$, where the first component is longitudinal displacement, the second component is longitudinal velocity and the third component is the engine torque applied at the wheels. The output measurement is $y_k \in \mathbb{R}^3$, and incorporates longitudinal displacement, longitudinal velocity, and acceleration measured from the on-board accelerometer. The output is a discrete time signal indexed by $k \in \mathbb{N}$ with constant time-step $\Delta T > 0$, where the correspondence to time t is given by $t = k\Delta T$. The process dynamics are given by

$$\dot{\hat{e}}(t) \quad = \quad A(q(t))\hat{e}(t) + B(q(t))u(t) + w(t), \quad y_k = C_k\hat{e}(k\Delta T) + D_k u(k\Delta T) + v_k,$$

where $w(t) \sim (0, Q)$ is continuous-time white noise with covariance Q, and $v_k \sim (0, R)$ is discrete-time white noise with covariance R.

Let the matrix $P(t)$ denote the estimated state error covariance, which is initialized to the identity matrix. Then, the prediction step of the filter is given by the following update equations, which represent a forward Euler approximation of the continuous time dynamics

$$\begin{aligned} \hat{e}(t) &= \hat{e}(t^-) + t_\Delta(A(q(t))\hat{e}(t^-) + B(q(t))u(t)) \\ P(t) &= P(t^-) + t_\Delta(A(q(t))P(t^-) + P(t^-)A(q(t))^T + Q), \end{aligned}$$

where t^- is the time of the previous update, and $t_\Delta := t - t^-$. A prediction step is performed every time the software system updates the current state, therefore, in general the time-step t_Δ is not constant. The correction step occurs only when a new longitudinal state measurement y is available and consists of the following update equations

$$\begin{aligned} K_k &= P(t^-)C^T(CP(t^-)C^T + R)^{-1} \\ \hat{e}(t) &= \hat{e}(t^-) + K_k(y_k - (C\hat{e}(t^-) + Du(t))) \\ P(t) &= (I - K_kC)P(t^-)(I - K_kC)^T + K_kRK_k^T. \end{aligned}$$

By nature of the fixed rate of measurements (discrete-time) and continuous-time inputs, the filter is said to be hybrid [87].

The matrices A, B, C, and D, have been identified from data for every gear q employing the system identification toolbox within MATLAB. In particular, we used a gray-box technique, where the system identification determines a vector of parameters, given a matrix structure derived from first principles. In particular, we have a second order system with rolling friction and inputs. We assume a multiplicative gear ratio from engine input to change in wheel torque. Therefore, the matrices are of the following form

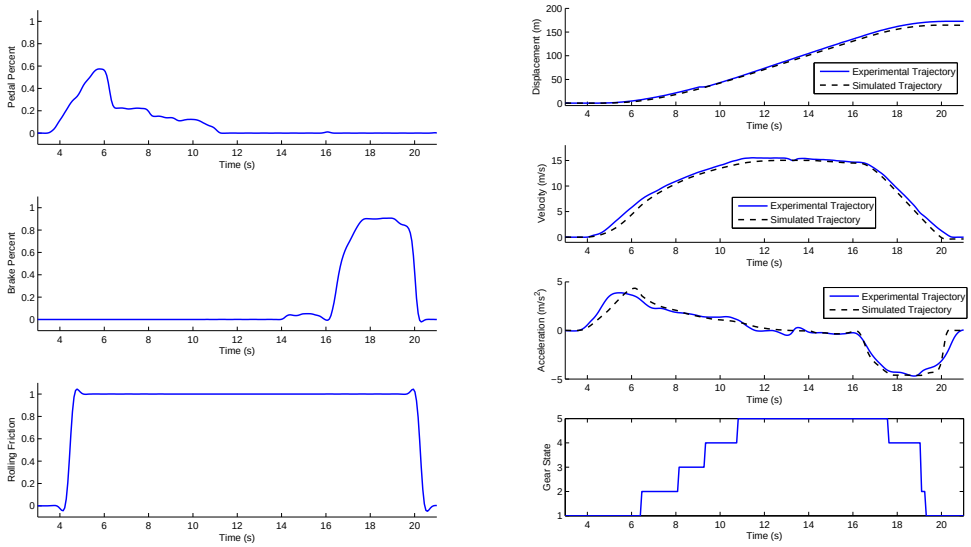
$$A(q) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & a(q) \end{bmatrix}, \quad B(q) = \begin{bmatrix} 0 & 0 & 0 \\ b_1 & 0 & b_2 \\ 0 & \alpha(q)b_3(q) & 0 \end{bmatrix},$$
$$C(q) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad D(q) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_1 & 0 & \alpha(q)b_3(q) \end{bmatrix}.$$

Data to preform this identification task was taken from four driving trials with varying input signals. The input signals were chosen by the driver to ensure an adequate sweep of the vehicles dynamic range under consideration. Each trial was taken on the path for which the vehicle normally drives on.

From the experimental data collected, we obtained for $q = 1$ that $a(q) = -2.5$, $b_1 = -5$, $b_2 = -0.1$, $b_3(q) = 5$, and $b_1 = 0.002$. For $q \in \{2, 3, 4, 5, 6\}$, we obtained that $a(q) = -1$, $b_1 = -5$, $b_2 = -0.1$, $b_3(q) = 5$, and $b_1 = 0.002$. The gear ratios are given by $\alpha(1) = 3.5$, $\alpha(2) = 2.0$, $\alpha(3) = 1.5$, $\alpha(4) = 1.2$, $\alpha(5) = 1$, and $\alpha(6) = 0.8$, which were taken from a technical data sheet [4]. This model was validated by comparing simulations obtained with an experimental input signal with the experimental trajectories, as seen in Figure 4.7.

The order preserving properties of this model with respect to input and state can be

verified for piecewise affine systems by checking Hypothesis 1 of [19], which is satisfied for this system because each matrix $A(q)$ satisfies the classical Kamke-Muller conditions (all off diagonal entries are positive).



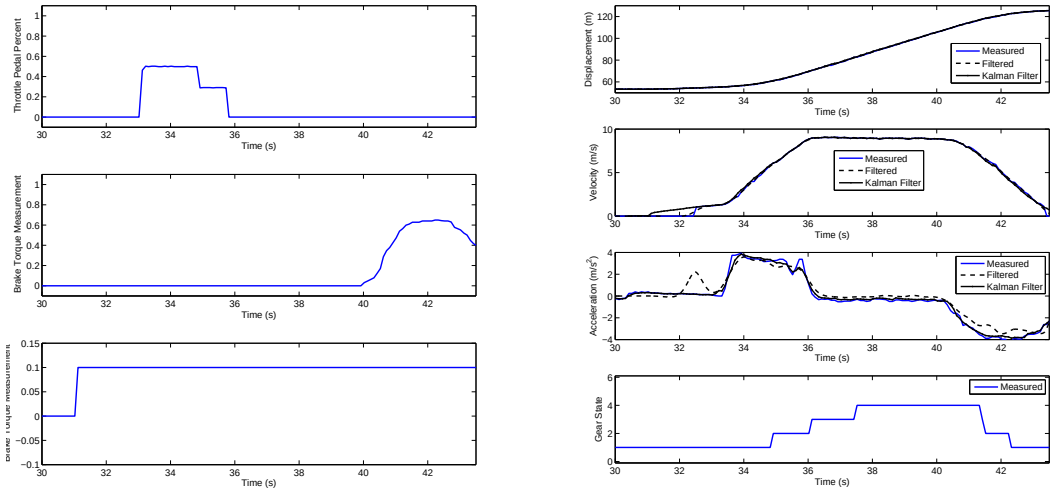
(a) (b)

Figure 4.7: Experimental trial used to identify the system dynamics, along with validate the resultant model. Open-loop simulation is performed using the learning system data. In (a), the experimental input signals for brake pedal percent, throttle pedal percent and rolling friction are provided. In (b), experimental output is compared to an open loop simulation using the identified model. The simulated trajectory is generated by the nominal initial conditions, and the input signals shown in (a).

To implement the Kalman filter, we chose the process and output noise covariance matrices to maximize noise rejection while still maintaining satisfactory bandwidth. We assume all noise processes are independent and identically distributed and have no mode dependency, therefore, the covariance matrices are all diagonal. The matrices are given as $R = \text{diag}(0.5, 0.3, 1)$ and $R = \text{diag}(0.5, 1, 1)$.

Kalman Filter performance is demonstrated in Figure 4.8, where we compare raw measurement data, post-processed filtered data, and the Kalman Filter state estimate. The post-processed filtered data is constructed by differentiated cubic smoothing spline fit to

raw velocity measurements in MATLAB.



(a) (b)

Figure 4.8: Experimental data comparing the closed loop Kalman filter, the raw measurements, and filtered off-line data. The offline data is the measurement data filtered, and is assumed to be the actual state.

The Kalman filter is used to construct a state prediction. This is accomplished by computing the *acceleration profile* $\mathcal{A}_{\bar{t}}$, a set-valued signal containing all possible acceleration trajectories for future times $t \geq \bar{t}$. This allows to predict the set of possible speeds $\hat{e}_2(t)$ for $t \geq \bar{t}$. Mathematically, this is given as $\hat{e}_2(t) \in \hat{e}_2(\bar{t}) + \int_{\bar{t}}^{t} \mathcal{A}_{\bar{t}}(\tau)d\tau$.

By the order preserving property of the dynamics with respect to input, we can represent the set $\mathcal{A}_{\bar{t}}$ as an interval set evolving in time, that is $\mathcal{A}_{\bar{t}}(t) \subset [l(t), h(t)]$, where $l(t), h(t) \in \mathbb{R}$. To compute the upper and lower bounds in practice, the predictive capability of the Kalman Filter is utilized with the time-step $t_\Delta = \Delta T$ (Section 4.5.1), assuming constant discrete state $q^{\bar{t}}(t) := q(\bar{t})$ for all $t \geq \bar{t}$. Denote the upper flow $\vee\hat{e}(t)$ and the lower flow $\wedge\hat{e}(t) \in X$. The initial condition chosen as $\wedge\hat{e}_0 = \vee\hat{e}_0 = \hat{e}(\bar{t})$. The upper and lower bounds of the acceleration profile can be computed with extremal inputs by virtue of the order preserving properties the flow with respect to input (Section 4.5.1). The lower bound, defined as the

sequence l_k, is computed with the minimum input $u_{min} \in \mathcal{U}$ as

$$\begin{aligned} \wedge\hat{e}_k &:= \wedge\hat{e}_{k-1} + \Delta T(A(q(\bar{t})) \wedge \hat{e}_{k-1} + B(q(\bar{t}))u_{min}), \\ l_k &:= [0\ 0\ 1](C \wedge \hat{e}_k + Du_{min}), \end{aligned}$$

where $u_{min} = (80, 0)$. The upper bound, given by the sequence h_k, is computed with the maximum input as

$$\begin{aligned} \vee\hat{e}_k &:= \vee\hat{e}_{k-1} + \Delta T(A(q(\bar{t})) \vee \hat{e}_{k-1} + B(q(\bar{t}))u_{max}), \\ h_k &:= [0\ 0\ 1](C \vee \hat{e}_k + Du_{max}), \end{aligned}$$

where $u_{max} = (0, 100)$. The discrete time sequences are converted to continuous time signals using a zero-order hold approximation, giving the acceleration profile

$$\mathcal{A}_{\bar{t}}(t) = \sum_k [l_k, h_k](H((t-\bar{t}) - (k-1)T_p) - H((t-\bar{t}) - kT_p)), \tag{4.14}$$

where $t \geq \bar{t}$ and $H(t)$ denotes the heavy side step function [78]. An example is provided as Figure 4.9. As mentioned in Section 4.3.3, Algorithm 2 requires a two-vehicle state

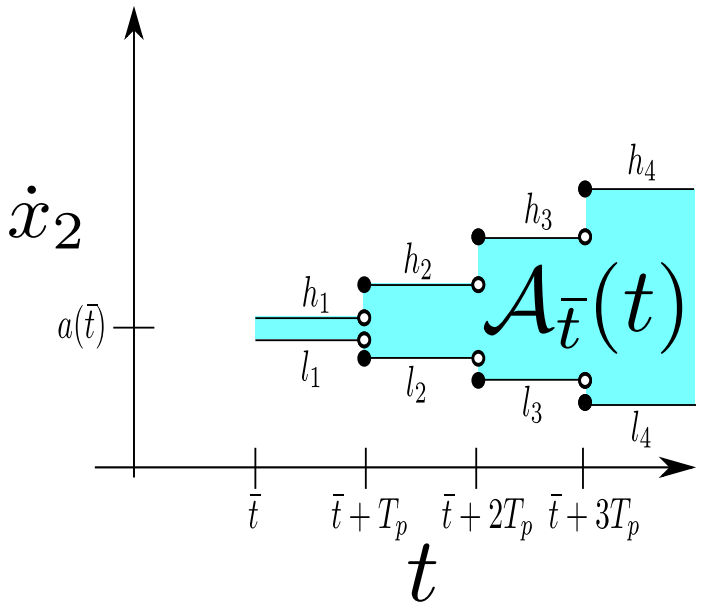


Figure 4.9: Acceleration profile $\mathcal{A}_{\bar{t}}$ example.

prediction, which has a tunable time-step Δ_p, which can be chosen by the test engineer, assumed to be less than 1.5 sec in total. With such a short time scale, it is reasonable to assume the input stays constant, that is $u(t) = u(\bar{t})$ for all $t \geq \bar{t}$. To account for the error of this assumption, we add a configurable window parametrized by the parameter $\beta \in \mathbb{R}_+$ to the resulting acceleration. As β is taken to 0, the prediction is assumed to exact. The calculation is carried out, to obtain upper and lower bound sequences $[l_k, h_k]$, with the Hybrid Kalman filter as

$$\begin{aligned} \hat{e}_k &= \hat{e}_{k-1} + \Delta T(A(q(\bar{t}))\hat{e}_{k-1} + B(q(\bar{t}))u(\bar{t})), \\ [l_k, h_k] &= [0\ 0\ 1](C\hat{e}_k + Du(\bar{t})) + k[-\beta, \beta], \end{aligned}$$

where set addition is understood in the sense of the Minkowski sum. The acceleration profile $\mathcal{A}_{\bar{t}}(t)$ is found by taking the zero-order hold approximation of the sequence $[l_k, u_k]$.

Full state estimator

The Kalman filter output is the estimate of position and speed, which are the first two components of $\hat{e}$, denoted by x^L for the local vehicle and by x^R for the remote vehicle, the estimate of global time t, and the acceleration profile $\mathcal{A}_{\bar{t}}(t)$. The full state estimate is constructed by combining local state estimation from the Kalman filter with received remote vehicle state information. In accordance with feedback map $g(\hat{x})$, as defined in Algorithm 2, evaluating control involves discretizing the flow and constructing the current state estimate $\hat{x}[n]$ and a prediction $\hat{x}[n+1]$. We now define the algorithm for computing the full state estimate and prediction, with arguments local state information $(x^L, t, \mathcal{A}^L_{\bar{t}^L})$, remote state information $(x^R, t^R, \mathcal{A}^R_{\bar{t}^R})$, and prediction time-step Δ_P. The state estimate is found with $FullStateEstimate$, defined in Algorithm 3,which returns the current state estimate $\hat{x}[n]$ and state prediction estimate $\hat{x}[n+1]$.

Algorithm 7 $(\hat{x}[n], \hat{x}[n+1])$ = FullStateEstimate$(x^L, x^R, t, t^R, \Delta_P, \mathcal{A}^L_{\vec{t}^L}, \mathcal{A}^R_{\vec{t}^R})$

Input: $(x^L, x^R, t, t^R, \Delta_P, \mathcal{A}^L_{\vec{t}^L}, \mathcal{A}^R_{\vec{t}^R}) \in 2^{X^L} \times 2^{X^R} \times \mathbb{R}^3_+ \times S(2^{\mathbb{R}}) \times S(2^{\mathbb{R}}) \times \mathbb{R}_+$

Synchronize remote state due to transmission delay
$\hat{x}^R_1[n] = x^R_1 + (t - t^R)x^R_2, \quad \hat{x}^R_2[n] = x^R_2 + (t - t^R)[\inf \mathcal{A}^R_{\vec{t}^R}(t^R - \vec{t}^R), \sup \mathcal{A}^R_{\vec{t}^R}(t^R - \vec{t}^R)]$
$\hat{x}[n] = x^L \times \hat{x}^R_1[n] \times \hat{x}^R_2[n]$

Construct prediction
$\hat{x}^L_1[n+1] = \hat{x}^L_1[n] + \Delta_p \hat{x}^L_2[n], \quad \hat{x}^L_2[n+1] = \hat{x}^L_2[n] + \Delta_p[\inf \mathcal{A}^L_{\vec{t}^L}(t - \vec{t}^L), \sup \mathcal{A}^L_{\vec{t}^L}(t - \vec{t}^L)]$
$\hat{x}^R_1[n+1] = \hat{x}^R_1[n] + \Delta_p \hat{x}^R_2[n], \quad \hat{x}^R_2[n+1] = \hat{x}^R_2[n] + \Delta_p[\inf \mathcal{A}^R_{\vec{t}^R}(t - \vec{t}^R), \sup \mathcal{A}^R_{\vec{t}^R}(t - \vec{t}^R)]$
$\hat{x}[n+1] = \hat{x}^L_1[n+1] \times \hat{x}^L_2[n+1] \times \hat{x}^R_1[n+1] \times \hat{x}^R_2[n+1]$

Output: $(\hat{x}[n+1], \hat{x}[n]) \subset 2^X \times 2^X$.

4.5.2 Communication

The state prediction performed by the estimator is necessary to account for communication delays and avoid control to be evaluated on old information. Communication delay comprises all delay experienced from the instant measurement data is populated on-board the local vehicle until the remote vehicle uses this state information to construct a capture set for control evaluation. This can be broken down into the following major components: (1) ICA application acquisition of state information from the local state estimator; (2) construction of a remote data message as commanded by the ICA application; (3) interface with communication layer Denso WSU radio; (4) physical delay in the wireless transmission of the information; (5) reception of the message from the remote vehicle communication layer; (6) population of this state information into the ICA application for use in capture set construction and subsequent control evaluation. From experimental results, we have found that the worst case delay is 0.4 seconds. Hence the multiple predictions performed to determine $\hat{x}[n+1]$ are such that the time $\Delta_p \approx 0.4$ seconds.

4.5.3 Control

The set-valued feedback map g is computed locally on each vehicle. To accommodate delay in the system arising from communication, software and actuators (as discussed

before, we evaluate the feedback controller for a set of state estimate predictions. Let the state estimate $\hat{x}[n]_i \subset X$ represent the estimate on-board vehicle i at time t. Algorithm 3 can be used recursively to construct more state estimate predictions. Define the prediction horizon count $N_p \in \mathbb{N}$, which is a configurable design parameter. We construct the state estimate predictions on-board vehicle i, given by $\hat{x}[n+j]_i$ for $1 \leq j \leq N_p$, as follows $(\hat{x}[n+j]_i, \hat{x}[n+j-1]_i) = \text{FullStateEstimate}(\hat{x}[n+j-1]_i, t + j\Delta_p, t^R + j\Delta_p, \Delta_p, \mathcal{A}^L_{t^L}, \mathcal{A}^R_{t^R})$, where the local vehicle refers to vehicle $i \in \{1, 2\}$. We then use the set of predictions to evaluate the feedback map g on-board vehicle $i \in \{1, 2\}$, implemented as $g(\hat{x}[n]_i) := \bigcap_{1 \leq j \leq N_p} \text{FeedbackMap}(\hat{x}[n+j]_i, \hat{x}[n]_i)$.

Before applying control, the two vehicles should reach an agreement on the control commands to apply. In general, we have that $\hat{x}[n]_1 \neq \hat{x}[n]_2$. However, both sets contain the true system state x by construction. As a consequence, we have that $g(\hat{x}[n]_i) \subseteq g(x)$ given the order reversing property of the map g. As a consequence, we can take $g(\hat{x}[n]_1) \cup g(\hat{x}[n]_2)$ as the set of all possible safe control choices. In practice, we implement this with a handshake mechanism to guarantee that both vehicles choose the same actions. Specifically, the handshake module remains in the trivial initial state until a collision is predicted on-board the local vehicle. From Algorithm 2, a collision is predicted on-board vehicle i when $g(\hat{x}[n]_i) \neq \mathcal{U}$, at which point a message is sent to the remote vehicle indicating a collision has been predicted. Vehicle i then waits for a message indicating a collision has been predicted on-board the second vehicle j. If no such message is received, the application sleeps for 10 ms and then re-sends the message denoting a collision has been predicted (in case the message was not received). This process continues until a message has been received from vehicle j, or it times out. If a message is received, then a consensus control is chosen and applied to the local actuator of both vehicles.

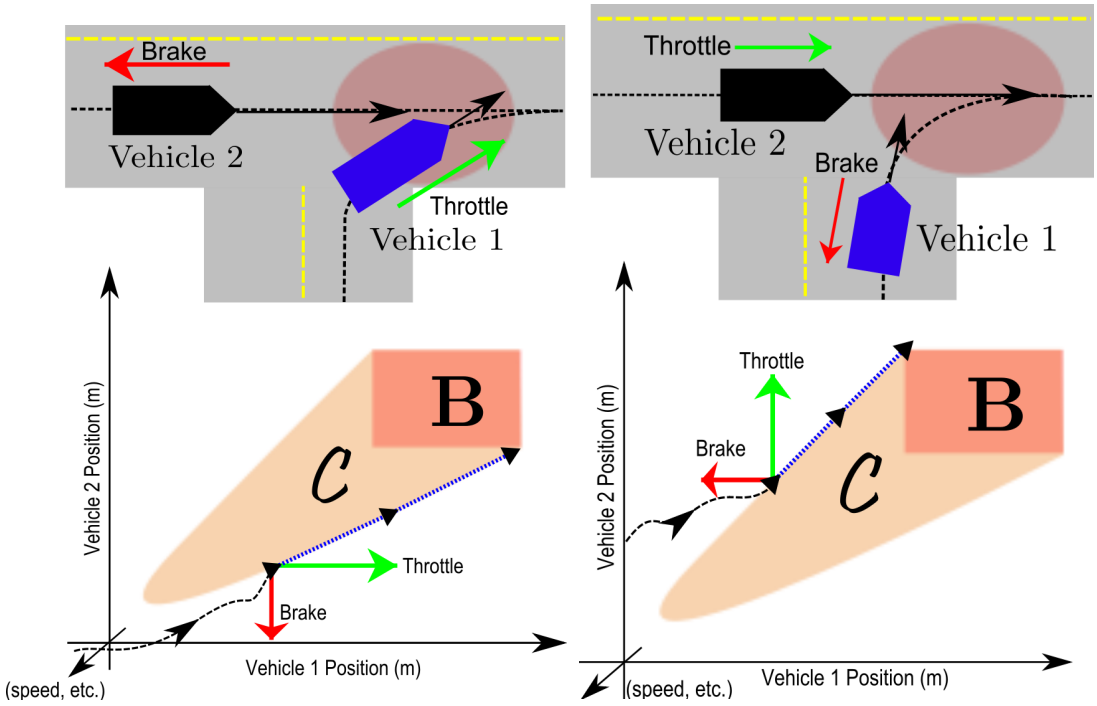


Figure 4.10: (a) Use case A involves a merging vehicle entering the intersection without first checking oncoming traffic. The figure shows a top down cartoon of this scenario along with the system configuration related to the capture set in the position plane X_1 for a fixed pair of vehicle speeds. (b) Use case B involves a merging vehicle approaching the intersection while misjudging the speed of oncoming traffic. The figure shows a top down cartoon of this scenario along with the configuration of the system related to the capture set in the X_1 plane.

4.6 Intersection Collision Avoidance Experiments

4.6.1 Experiment Setup

Experiments were conducted at the TEMA test track in Ann Arbor, Michigan employing two modified Lexus IS 250 vehicles (Figure 4.1(c)). Both vehicles run ICA as they approach the intersection. The velocity of approach is not fixed, however it must be within safe limits. Each path is stored as a list of UTM co-ordinates on the respective vehicle. The speed limits for path 1 are $v_{min} = 0$ m/s and $v_{max} = 8.8$ m/s, while the speed limits for path 2 are $v_{min} = 8.8$ m/s and $v_{max} = 18$ m/s. The bad set parameters chosen are $L^1 = 55$ m, $L^2 = 75$ m, $H^1 = 65$ m and $H^2 = 85$ m. The input sets are chosen to be $\mathcal{U}^1 := [u_L^1, u_H^1] = [0, 0.3] \times [0, 0.5]$ and $\mathcal{U}^2 := [u_L^2, u_H^2] = [0, 0.3] \times [0, 1]$, which represent

extremal inputs that maintain safe driving conditions along the vehicle paths within the speed limits.

We consider two real-world scenarios, which we refer to as "use cases". For use case A, we assume a merging vehicle enters the intersection without properly surveying for oncoming traffic. Since the vehicle has already entered the intersection (or the speed is too high such that this is unavoidable), the only solution is for the merging vehicle to apply throttle and the straight vehicle to brake. A visualization of this is provided in Figure 4.10(a). For use case B, we assume a merging vehicle is approaching an intersection at high speed, and likely misjudging the speed of oncoming traffic. The solution in this case is for the merging vehicle to apply brake while the straight vehicle applies the throttle. A visualization of this is provided in Figure 4.10(b). We performed a total of 28 trials, 15 for use case A and 13 for use case B.

4.6.2 Experiment results

All trajectories generated by the experiments are provided in Figure 4.11 in the displacement plane. As it is apparent from the plots, no trajectory ever entered the bad set, hence all collisions were averted. Also, the trajectories pass fairly close to the bad set, indicating that the control algorithm is non- conservative as expected from theory. In order to better quantify the performance, we calculated the distance of the trajectory of the system from the capture set, denoted γ, and the distance of the trajectory from the bad set, denoted ζ. Table 4.1 provides the summary of the results. This table shows that the trajectory never entered the capture set during any trial. This is expected from theory as the controller guarantees that trajectories starting outside of the capture set remain outside of the capture set. Furthermore, the distances of the trajectories from the capture set are very small and can be decreased by decreasing the prediction horizon Δ_p and removing the state uncertainty

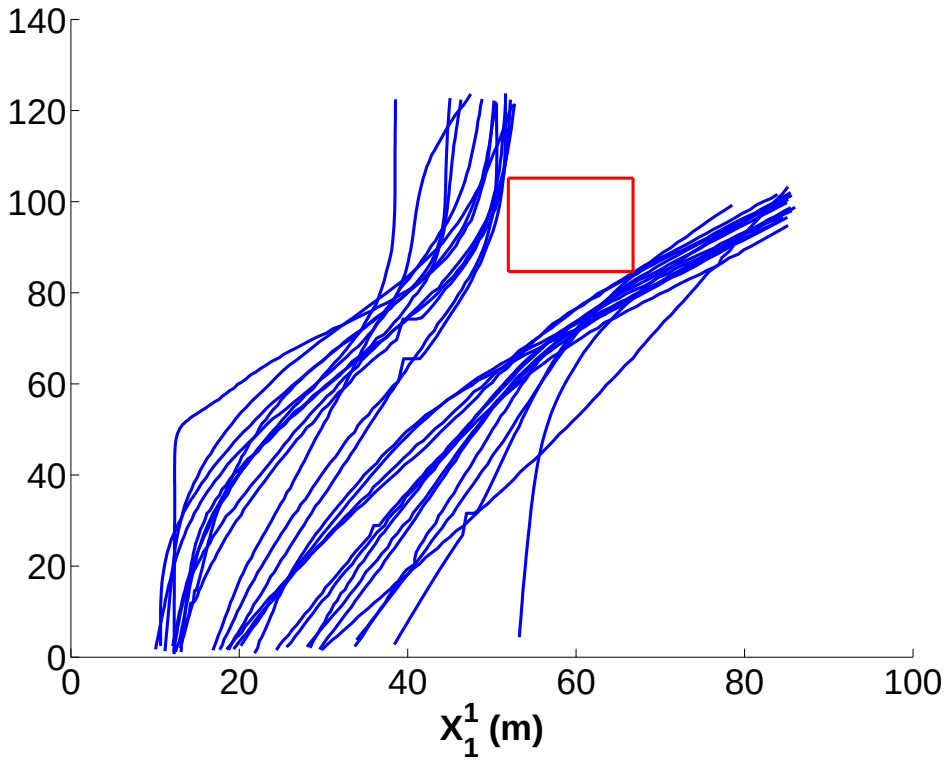


Figure 4.11: All trajectories from all trials. The safety specification is maintained given that the flow of the system never entered the bad set **B** during any trial.

β. With no state uncertainty ($\beta = 0$), the trajectories pass extremely close to the capture set and to the bad set, indicating a very aggressive and non-conservative controller. When state uncertainty is introduced, the distances of the trajectory from the capture set and from the bad set increase, but they can be rendered smaller by decreasing the prediction horizon Δ_p used at each state prediction step. For a small prediction horizon, even with a larger number of prediction steps N_p, the distances from both the capture set and the bad set are still fairly small indicating that the controller is non-conservative. Our algorithms hence also provide a number of design parameters to compromise how aggressive the controller is (measured by how close to the bad set the trajectories go) with the control conservatism (the controller acts sooner than it could have). This tradeoff is very important in practice because overriding the driver can be justified only if it is needed to keep the system safe.

# Trials	N_p	Δ_p	Info	ζ (m) min/avg	γ (m) min/avg	Entered C	Entered **B**	Use case
4	3	0.4	P	0.9, 3	0.7 , 2.8	No	No	A(2), B(2)
4	4	0.2	P	0.6, 0.9	0.1, 0.6	No	No	A(2), B(2)
14	3	0.4	I	2, 5.9	2, 5.8	No	No	A(9), B(5)
6	4	0.2	I	0.7, 1.7	0.5, 1.4	No	No	A(2), B(4)

Table 4.1: The first column indicates the number of trials, the second column the number of prediction steps N_p employed for evaluation the control map (Section 4.5.3), Δ_p is the prediction time employed in state prediction (Algorithm 3),"P" denotes perfect state information ($\beta = 0$ in the prediction step of Section 4.5.1) and "I" denotes imperfect state information ($\beta = 0.2$), ζ and γ are the distances of the trajectory from the bad set **B** and from the capture set C, respectively. We show both the minimum value and the average value across the trials. The trajectory never entered the capture set nor the bad set in any trial.

Our results show that the controller, while being robust to un-modeled dynamics, state uncertainty, and communication delays, can be tuned so to override the drivers only when it is necessary to prevent a collision.

Figure 4.12 shows an experimental trial with perfect state information ($\beta = 0$) and with use case A, while Figure 4.13 shows a trial for use case B and imperfect state information ($\beta \neq 0$). In use case A (Figure 4.12), the merging vehicle (vehicle 1) approached the intersection at a cruising speed of 6 m/s, while vehicle 2 approached the intersection at an accelerating speed of around 14 m/s. To avoid the collision, the drivers were overridden at time 19.7 sec when the state prediction hit the boundary of the capture set. At this time, automatic throttle was applied to vehicle 1 and automatic brake was applied to vehicle 2. This control results in vehicle 2 entering the intersection only (and immediately) after vehicle 1 has cleared the intersection. Vehicle 1 reached the speed limit v^1_{max} while applying throttle, after which time, the controller held the speed constant. The test ended after the merging vehicle exited the intersection, after which time, automatic control was deactivated and the driver retained control. While conducting this experiment, the system trajectory $\hat{x}(t)$ was at least within 0.7 m of the capture set, while never actually entering it, which implies safety was maintained and the control actions were not conservative.

In use case B (Figure 4.13), imperfect state information was considered using $\beta =$

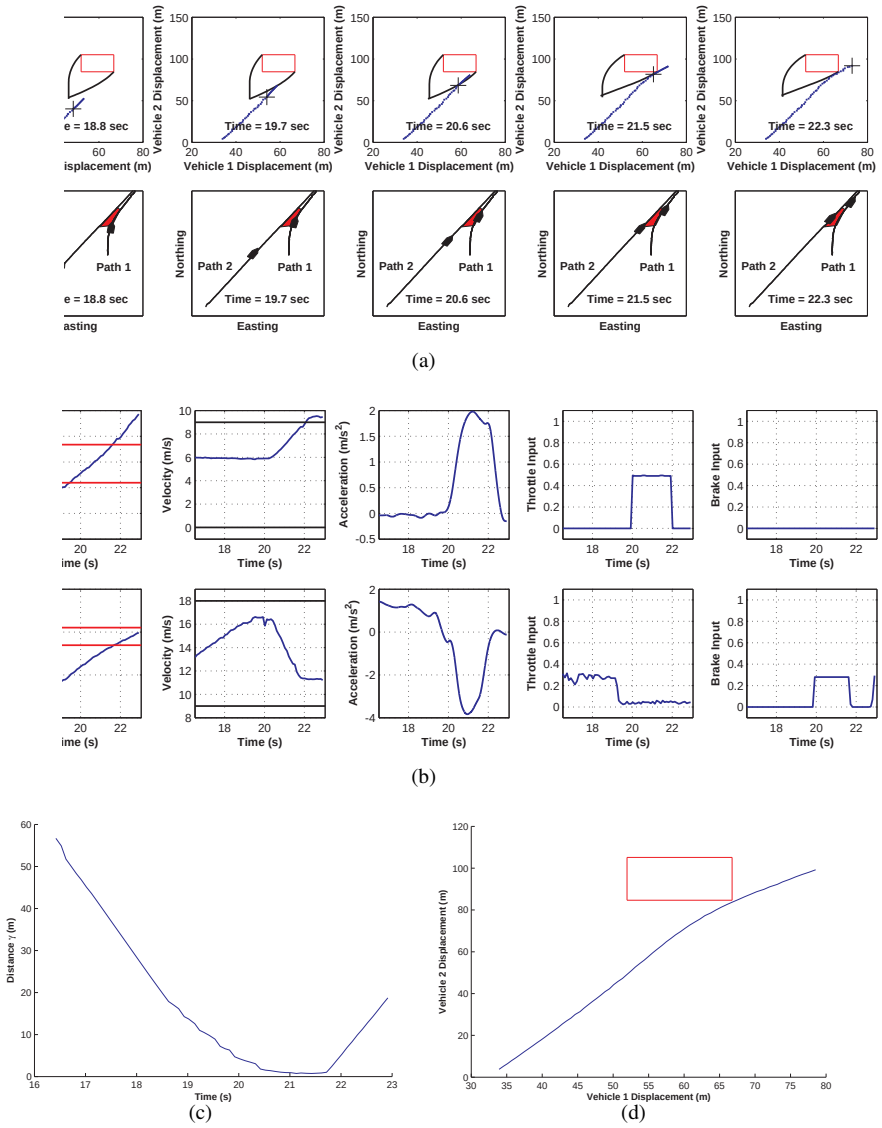


Figure 4.12: An experimental trial for use case A. Here, perfect state information is assumed. (a) Snapshots showing the configuration of the vehicles at different times. The upper row shows the configuration of the vehicles (indicated by the cross) in the displacement space along with the capture set slice C (delimited by the black line) corresponding to the current vehicle speeds. The bad set is the red box. The solid blue line indicates the trajectory in the displacement space. The portion of this line ahead of the cross indicates the state prediction. The lower row shows the vehicle positions as they appear from a top-down view of the experiment. The red area corresponds to the bad set (red box in the upper row plots). (b) Signals for vehicle 1 are shown in the upper row, while the bottom row shows signals for vehicle 2. At time 19.7 sec, the state prediction hits the boundary of the capture set and hence vehicle 1 applies throttle and vehicle 2 applies brake. (c) Distance between state and capture set shown as a function of time. (d) Entire trajectory for the test.

0.2 m/s^2. In this trial, the merging vehicle (vehicle 1) started at rest, while vehicle 2 approached the intersection at an accelerating speed of around 8 m/s. Vehicle 1 attempted to violently accelerate and enter the intersection. To avoid the collision, the drivers were

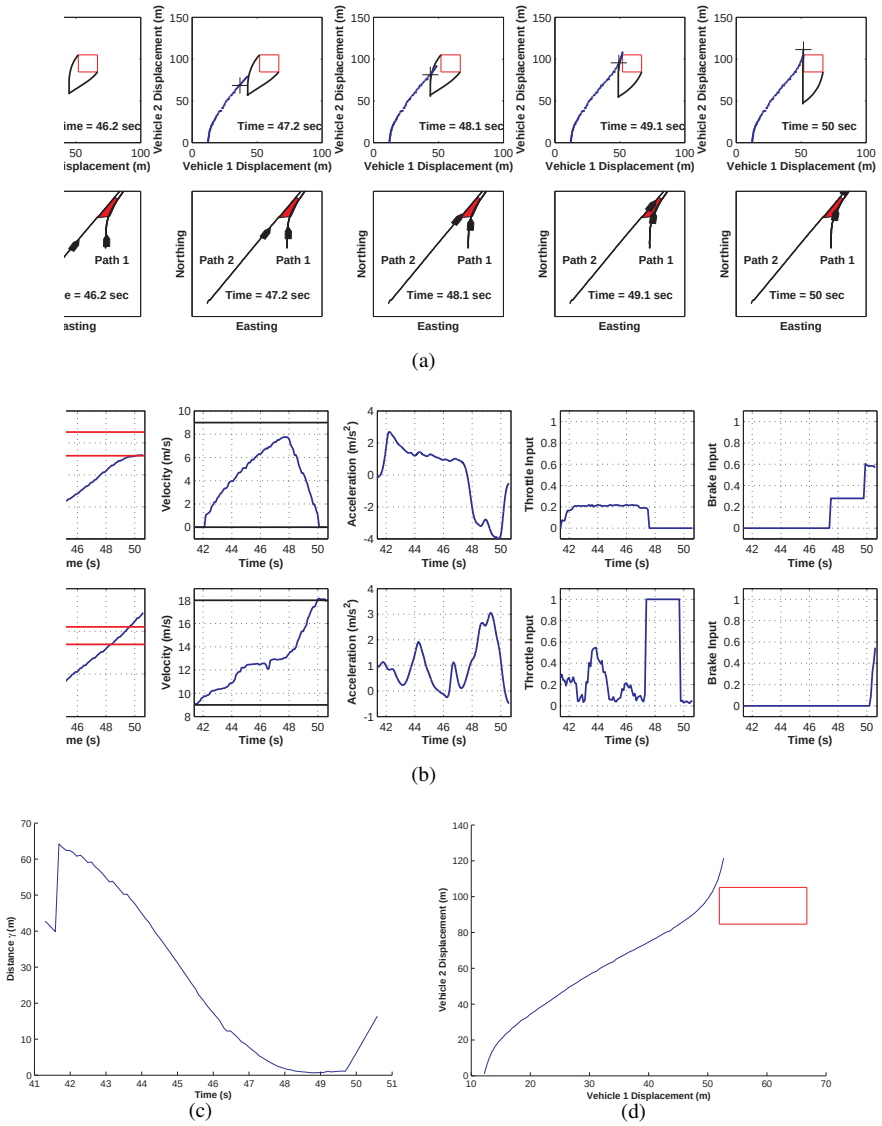


Figure 4.13: An experimental trial for use case B. Imperfect state information is considered here ($\beta \neq 0$). The upper row shows the configuration of the vehicles (indicated by the cross) in the displacement space along with the capture set slice C (delimited by the black line) corresponding to the current vehicle speeds. The bad set is the red box. The solid blue line indicates the trajectory in the displacement space. The portion of this line ahead of the cross indicates the state prediction set. In this experiment, $N_p = 3$ and $\Delta_p = 0.4$ and the resulting uncertainty in position is very small (about 0.1 m), so it is hardly visible in the plot. However, the uncertainty on the speed is significant and it is about 0.5 m/sec. The velocity signal displays the estimate velocity x_2^I resulting from the Kalman filter. The lower row shows the vehicle positions as they appear from a top-down view of the experiment. The red area corresponds to the bad set (red box in the upper row plots). (b) Signals for vehicle 1 are shown in the upper row, while the bottom row shows signals for vehicle 2. At time 47.2 sec, the state prediction hits the boundary of the capture set and hence vehicle 2 applies throttle and vehicle 1 applies brake. (c) Distance between flow and capture set shown as a function of time. (d) Entire trajectory for the test.

overridden at time 47.2 sec, when the set prediction hit the boundary of the capture set. In this case, automatic brake was applied to vehicle 1 and automatic throttle was applied to

vehicle 2. This control results in vehicle 1 entering the intersection only (and immediately) after vehicle 2 has cleared the intersection. The merging vehicle reached the speed limit v^1_{min} while applying brake, after which time, the controller held the vehicle at rest. The straight vehicle reached the speed limit v^2_{max} while applying throttle, after which time, the controller held the vehicle at a constant speed. The test ended after the straight vehicle exited the intersection, after which time, automatic control was deactivated and the driver retained longitudinal control. While conducting this experiment, the system trajectory $\hat{x}(t)$ was within 0.6 m of the capture set, while never actually entering it, which implies safety was maintained and the control actions were not conservative.

CHAPTER V

Conclusion

In this thesis we developed a general formulation for the safety control problem encountered in multi-agent systems. The primary motivation was toward traffic networked systems, where problems of imperfect sensor information, communication delay, and computational limitations make traditional centralized control approaches prohibitive. This is a practical problem without a well defined solution, where fully automated traffic networks are not feasible these days due to both legal and commercial hurdles. Therefore, it is advantageous to develop automatic control systems that can be implemented in parallel with human drivers, such as cases where control is applied only when it is absolutely necessary. Furthermore, formal methods are attractive in the sense that safety can be shown mathematically rather than through exhaustive ad-hoc testing. While localized to the problem of two-vehicle collision avoidance, the methods developed in this thesis will function as a building block for more complex autonomous and semi-autonomous systems.

5.1 Summary

Since the dynamic feedback problem for general hybrid systems with imperfect state information is prohibitive, we focused on a restricted class of systems, which is still relevant for modeling a number of application scenarios. In particular, we focused on a class of hybrid systems with order preserving dynamics. For this class of systems, we have pre-

sented an explicit solution to the safety control problem with imperfect state information. These results were presented under the assumption of disturbance inputs, which can represent un-modeled dynamics, or situations where one of the agents is uncontrolled. We have provided linear complexity discrete-time algorithms for computing this solution. We have shown the application of these algorithms to a two-vehicle collision avoidance scenario at a traffic intersection. The experimental results confirm the suitability of these algorithms for fast real-time computation.

Next, the problem of a three-vehicle roundabout system was considered, where the safety specification was defined as the union of all possible two-vehicle collisions. With this approach, we were able to construct feedback controllers using the primitives introduced in Chapter II, assuming both cases of cooperative and competitive control between the vehicles. A formal procedure for checking the non-blocking conjunction was developed, where we showed the system maintains safety. This was verified through experimental data taken in the multi-agent test-bed at the University of Michigan.

Lastly, we showed how these results could be extended and implemented on-board full-size test vehicles. In this case, the disturbance model was utilized to account for complexity in the vehicle dynamics, whose trajectories often did not follow deterministic trajectories. The method of system identification utilized simple field experimentation, and resulted in a model, while not conservative, which gave guaranteed dynamical performance. A set-valued state estimator was developed on the vehicles computer which could by design account for communication delay and synchronization errors. Such delay was found to be inherent in a distributed control system with a dynamic nature of this caliber. A control handshake mechanism was developed to account for the distributed evaluation of control.

Experimentally, we have shown how one can tune the prediction horizon and the num-

ber of prediction steps in order to tune the conservatism, that is, how soon the controller decides that automatic control is needed to prevent an imminent collision. The later the automatic control acts, the less conservative the algorithm is, but the closer the system trajectories come to a collision (while still averting it). This trade off can be decided depending on the system specifications.

The algorithms are guaranteed to be safe by design as they are based on calculating the capture set and on keeping the system state outside of the capture set. These algorithms can account for imperfect information due to sensor noise and to communication delays, need only a coarse model of the vehicle dynamics, can be efficiently run in real-time, and are guaranteed from a theoretical point of view to be the least conservative.

5.2 Future Work

5.2.1 General Classification for Piecewise Continuous Order Preserving Systems

The order preserving properties of a dynamical system are often compromised by the introduction of computer subsystems, as is the case for the transmission module used to describe the full vehicle system. While it is unclear what the order preserving properties of this system are in general, it is true that when the dynamics are abstracted to case of disturbance inputs, we do know that the convex hull of all possible trajectories generated by the differential inclusion (set of disturbance inputs) has certain order preserving properties. We aim to show that, under the more relaxed assumption

$$\phi(t, x, \mathbf{u}, \mathbf{d}_L) \leq \phi(t, x, \mathbf{u}, S(D)) \leq \phi(t, x, \mathbf{u}, \mathbf{d}_H) \; \forall \; t \in \mathbb{R}_{\geq 0}, \tag{5.1}$$

$$\phi(t, x, \mathbf{u}_L, \mathbf{d}) \leq \phi(t, x, S(U), \mathbf{d})) \leq \phi(t, x, \mathbf{u}_H, \mathbf{d}) \; \forall \; t \in \mathbb{R}_{\geq 0}, \tag{5.2}$$

then we can still say that $C = C_{\omega_{\mathcal{L}}} \cap C_{\omega_{\mathcal{H}}}$, and subsequently that the safety property of the system is maintained under the set-valued feedback $G : 2^X \rightrightarrows S(\mathcal{U})$.

Conditions for Envelope Properties

Rather than making an assumption on the flow of a dynamical system (5.1), it would be far more useful to establish this property from assumptions on piecewise vector fields with disturbances. Therefore, we aim to show that property (5.1), is a consequence of a vector field $f : X \times U \times D \to X$ that satisfies

$$f(x, u, d_L) \leq f(x, u, D) \leq f(x, u, d_H), \tag{5.3}$$

$$f(x, u_L, d) \leq f(x, U, d) \leq f(x, u_H, d). \tag{5.4}$$

Relationship between Abstract System Capture Set and Physical Vehicle Model Capture Set

If (5.3) is a sufficient assumption to conclude safety under the feedback controller introduced in (4.6), then we would like to establish safety of the system described in Section 4.4.1. This would follow if we can establish that the capture set generated by the vehicle system when considering the powertrain model.

5.2.2 Verify Modular Control Distributed

The idea of implementing controllers on a distributed manner is well understood in terms of state estimation, however problems arise when considering cooperative active control and handshaking. State estimates can always be generated more and more conservatively to account for delays, but differences between agents can lead to incompatible control actions. Therefore, control system models must account for the logic based decision trees for handshaking, which can be modeled as discrete event systems.

Handshaking

We design a handshake module that computes the logical disjunction of the local vehicle controller $g(\hat{x}^l(t, l))$ and the remote vehicle controller $g(\hat{x}^r(t, r))$. To formally describe the

action of the handshake mechanism, we introduce a discrete event formalism to model the software implementation of this module on each vehicle. To define safety of execution, we consider the parallel composition of the local and remote discrete event systems. Safety of the execution is then verified by using formal verification tools from the Discrete Event Systems literature [29].

Software Overview

This flowchart for the handshake module is given in Figure 5.1. The system remains in the trivial initial state until a collision is predicted on-board the local vehicle. From Algorithm 2.6.3, a collision is predicted when $g(X) \neq U$ on-board the local vehicle. Therefore, the state of the remote vehicle feedback controller is of no consequence if a collision is not predicted locally. Mathematically, the implication is that

$$g(\hat{x}^1(t,i)) \cup g(\hat{x}^2(t,i)) = U \cup g(\hat{x}^2(t,i)) = U,$$

implying no control is needed unless both vehicles predict a collision

To formally verify safety of the system when considering the handshake mechanism, we introduce a discrete event model to describe the handshake software module. The handshake module can be described by a discrete event system $H := \{X, E, f, \Gamma, x_0, X_m\}$, where $X \subset \mathbb{N}^3$ is a discrete state space, E is a finite set of events, $f : X \times E \to X$ is the transition function, $\Gamma : X \rightrightarrows E$ is the active event set-valued map, x_0 is the initial state, and $X_m \subset X$ are the marked states. The state space is defined by

$$\begin{aligned} X_L &:= \{\textbf{n}\text{ominal}, \textbf{w}\text{aiting}, \textbf{r}\text{ceieved}, \textbf{a}\text{ctive}, \textbf{c}\textit{ollision}\} = \{n, w, r, a, c\}, \\ X_2 &:= \{\textbf{s}\text{afe}, \textbf{t}\text{hrottle}, \textbf{b}\text{rake}, \textbf{a}\text{rbitrary}, \textbf{c}\text{ollision}\} = \{s, t, b, a, c\}, \end{aligned}$$

where X_1 denotes the *logic* state of the controller and X_2 denotes the *solution* state of the controller.

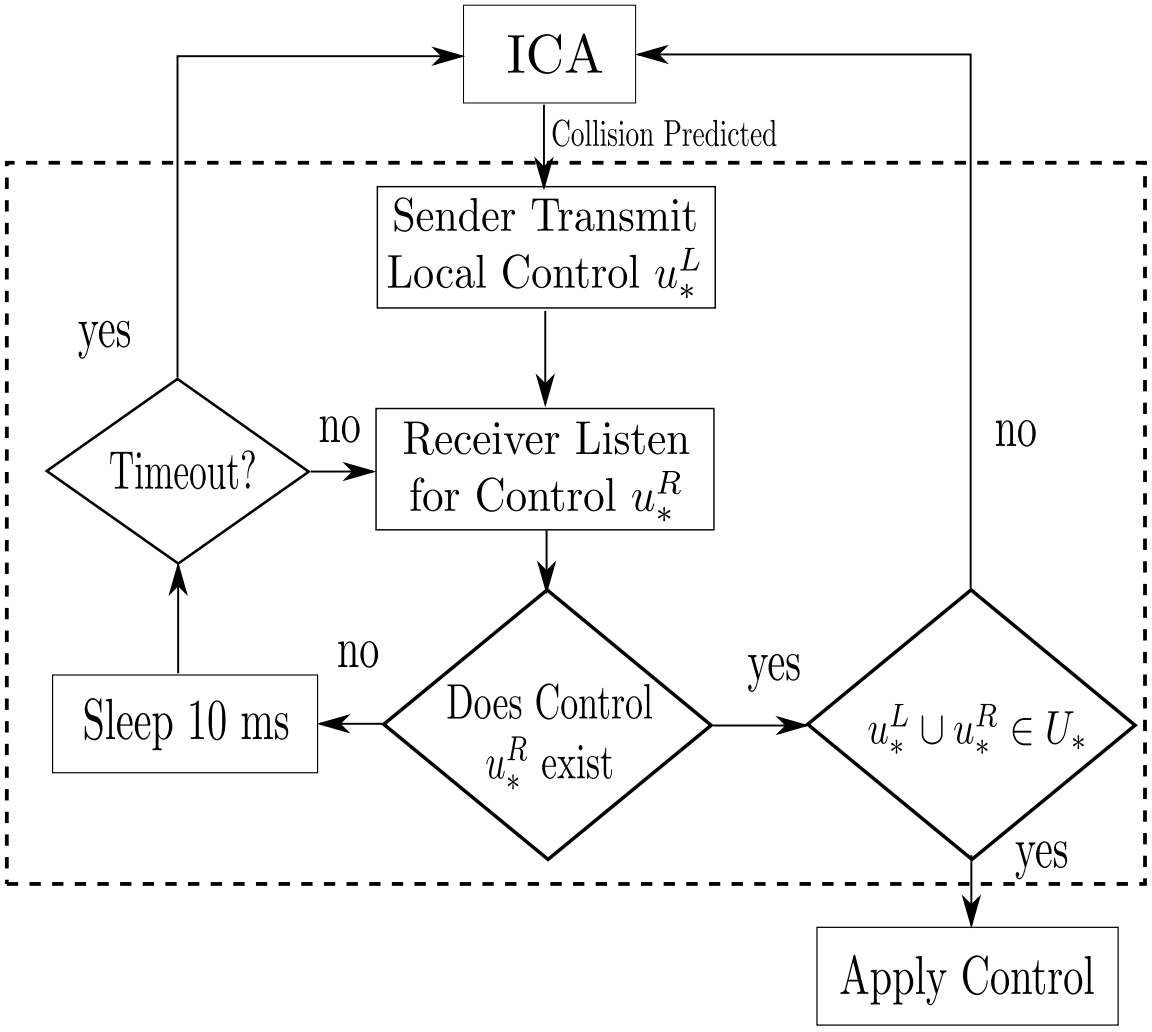


Figure 5.1: Handshaking overview.

Assume the logical state $x_1 \in \{r^L\} \times X_1^R$, define the handshake logic as follows between solution states $x_2^R \in X_2^L$ and $x_2^R \in X_2^R$ as

$$\mathcal{H}(x) := \begin{cases} (a, X_1^R, t^L, b^R) & \text{if } x_2 \in (t^L, b^R) \cup (a^L, b^R) \cup (t^L, c^R), \\ (a, X_1^R, b^L, t^R) & \text{if } x_2 \in (b^L, t^R) \cup (b^L, c^R), \\ (n, X_1^R, X_2) & \text{if } x_2 \in (b^L, b^R) \cup (t^L, b^R), \\ (c, X_1^R, c) & \text{if } x_2 \in (c^L, c^R). \end{cases}$$

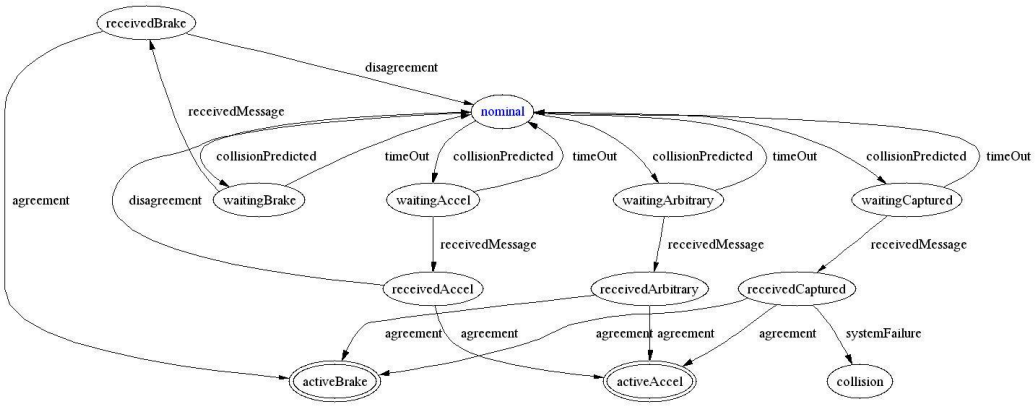


Figure 5.2: Handshaking automaton.

Safety Verification

The closed loop implementation of the handshake module between vehicles can be viewed as the parallel composition of two discrete-event system. We define the deadlock states $D \subset X$, where for $x \in D$, we have $\Gamma(x) = \emptyset$. This set represents the collection of all failure states, which in the considered application represents both the local and remote vehicle in the logical state ”collision predicted” with control state ”captured”. If this state within the two-vehicle automaton cannot be reached under certain language assumptions imposed on the individual vehicle models, then necessarily the system is provably safe under the abstracted logical formulation.

Extension to Timed Automaton

The continuous time control imposed by the supervisor $g(\hat{x}(t))$ can be abstracted away by means of imposing language restrictions on the automaton used to model the logical evaluation. That is, in order to arrive in the state $(collisionpredicted, captured)$, the state $(collisionpredicted, \{brake, throttle, arbitray\})$ must have been previously visited. We would like to extend this model to consider the timed execution of each step, since the handshake module must be implemented on-board a computer system.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Car 2 Car Communication Consortium. http://www.car-to-car.org.

[2] Cooperative Intersection Collision Avoidance Systems (CICAS). http://www.its.dot.gov/cicas.

[3] Crash Avoidance Metrics Partnership (CAMP). http://www.camp-ivi.com.

[4] http://www.vehix.com/car-reviews/2007/lexus/is-250/4dr-sport-sdn-auto-rwd/vehicle-specifications.

[5] Vehicle Infrastructure Integration Consortium (VIIC). http://www.vehicle-infrastructure.org.

[6] Vehicle Infrastructure Integration (VII). http://www.its.dot.gov/vii.

[7] National roundabout conference, trb. http://144.171.11.107/Main/Public/Blurbs/156622.aspx, 2005.

[8] http://www.itarda.or.jp/english/. 2011.

[9] Vehicle safety and fuel economy rulemaking and research priority plan 2011-2013. In *http://www.nhtsa.gov/staticfiles/rulemaking/pdf/2011-2013_Vehicle_Safety-Fuel_Economy_Rulemaking-Research_Priority_Plan.pdf*, 2011.

[10] A. Alessandri and P. Coletta. Design of observer for switched discrete-time linear systems. In *American Control Conference*, pages 2785–2790, 2003.

[11] M. Althoff. Reachability analysis and its application to the safety assessment of autonomous cars. In *PhD thesis, Technische Universitat Munchen*, 2010.

[12] L. Alvarez and R. Horowitz. Analysis and verification of the PATH AHS coordination-regulation layers hybrid system. In *American Control Conference*, pages 2460–2461, Albuquerque, New Mexico, 1997.

[13] L. Alvarez and R. Horowitz. Hybrid controller design for safe maneuvering in the PATH AHS architecture. In *American Control Conference*, pages 2454–2459, Albuquerque, New Mexico, 1997.

[14] L. Alvarez and R. Horowitz. Safe platooning in automated highway systems. *California Partners for Advanced Transit and Highways (PATH). Research Reports: Paper UCB-ITS-PRR-97-46.* `http://repositories.cdlib.org/its/path/reports/UCB-ITS-PRR-97-46`, Jan 1997.

[15] D. Angeli and E. D. Sontag. Monotone control systems. *Transactions on Automatic Control*, 48(10):1684 – 1698, 2003.

[16] D. Angeli and E.D. Sontag. Interconnections of monotone systems with steady-state characteristics. *Optimal control, stabilization and nonsmooth analysis. Lecture Notes in Control and Inform. Sci. Springer*, 301:135–154, 2004.

[17] D. Angeli and E.D. Sontag. Oscillations in I/O monotone systems. *IEEE Transactions on Circuits and Systems*, 55:166–176, 2008.

[18] E. Asarin, O. Maler, and A. Pnueli. Symbolic controller design for discrete and timed systems. *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Sceince, vol. 999, P. Antsakilis, W. Kohn, A. Nerode, and S. Sastry (Eds.), Springer Verlag, pages 1–20, 1995.

[19] A. Aswani and C. Tomlin. Monotone piecewise affine systems. *IEEE Transactions on Automatic Control*, 54(8):1913 –1918, 2009.

[20] J. Aubin. *Viability Theory*. Birkhäuser, 1991.

[21] J. Aubin and H. Frankowska. *Set-Valued Analysis*. Systems and Control: Foundations and Applications. Birkhäuser, Boston, 1990.

[22] J-P. Aubin, J. Lygeros, M. Quincampoix, S. Sastry, and N. Seube. Impulse differential inclusions: A viability approach to hybrid systems. *IEEE Transactions on Automatic Control*, 47:2–20, 2002.

[23] A. Balluchi, L. Benvenuti, M. D. Di Benedetto, and A. Sangiovanni-Vincentelli. Design of observers for hybrid systems. In *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science vol. 2289, C. J. Tomlin and M. R. Greensreet (Eds.), Springer Verlag, pages 76–89, 2002.

[24] A. Balluchi, L. Benvenuti, M. D. Di Benedetto, and A. Sangiovanni-Vincentelli. Observability of hybrid systems. In *Conf. on Decision and Control*, pages 1159–1164, 2003.

[25] A. Balluchi, L. Benvenuti, M.D. di Benedetto, C. Pinello, and A.L. Sangiovanni-Vincentelli. Automotive engine control and hybrid systems: challenges and opportunities. *Proceedings of the IEEE*, 88(7):888 –912, jul 2000.

[26] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, 45:1864–1876, 1999.

[27] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[28] M. S. Branicky. Multiple Lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):475–482, 1998.

[29] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer, 1999.

[30] A. Colombo and D. Del Vecchio. Efficient algorithms for collision avoidance at intersections. In *Hybrid Systems: Computation and Control*, 2012.

[31] E. F. Costa and J. B. R. do Val. On the observability and detectability of continuous-time jump linear systems. *SIAM Journal on Control and Optimization*, 41:1295–1314, 2002.

[32] B. A. Davey and H. A. Priesteley. *Introduction to Lattices and Order*. Cambridge University Press, 2002.

[33] D. Del Vecchio. Observer-based control for block-triangular hybrid automata. In *Conf. on Decision and Control*, 2007.

[34] D. Del Vecchio. A partial order approach to discrete dynamic feedback in a class of hybrid systems. In *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, vol. 4416, A. Bemporad, A. Bicchi, and G. Buttazzo (Eds.), Springer Verlag, pages 159–173, Pisa, Italy, 2007.

[35] D. Del Vecchio. Observer-based control of block triangular discrete time hybrid automata on a partial order. *International Journal of Robust and Nonlinear Control*, pages 1581–1602, 2008.

[36] D. Del Vecchio, M. Malisoff, and R. Verma. A separation principle for a class of hybrid automata on a partial order. In *American Control Conference*, pages 3638 – 3643, 2009.

[37] D. Del Vecchio, R. M. Murray, and E. Klavins. Discrete state estimators for systems on a lattice. *Automatica*, 42(2):271–285, 2006.

[38] D. DelVecchio. Discrete dynamic feedback for a class of hybrid systems on a lattice. In *Proc. of 2006 joint IEEE Conference on Control Applications, IEEE Computer Aided Control Systems Design Symposium, and IEEE International Symposium on Intelligent Control*, Munich, 2006.

[39] O. Maler E. Asarin and A.Pnueli. Symbolic controller synthesis for discrete and timed systems. In *Hybrid Systems II,* Lecture Notes in Computer Science, vol. 999, P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry (Eds.), Springer Verlag, pages 1–20, 1995.

[40] G.A. Enciso, H.L. Smith, and E.D. Sontag. Non-monotone systems decomposable into monotone systems with negative feedback. *J. of Differential Equations*, 224:205–227, 2006.

[41] European Road Safety Observatory (ERSO). Annual statistical report. 2006.

[42] Hosam K. Fathy, Rahul Ahlawat, and Jeffrey L. Stein. Proper powertrain modeling for engine-in-the-loop simulation. *ASME Conference Proceedings*, 2005(42169):1195–1201, 2005.

[43] Y. Gao, J. Lygeros, and M. Quincampoix. A viability approach to the reachability problem for uncertain hybrid systems. In *Mathematical Theory of Networks and systems*, 2004.

[44] Y. Gao, J. Lygeros, and M. Quincampoix. The reachability problem for uncertain hybrid systems revisited: A viability theory perspective. In *Lecture Notes in Computer Science LNCS, no. 3927*, pages 242–256, 2006.

[45] R. Ghaemi and D. Del Vecchio. Safety control of piece-wise continuous order preserving systems. In *Proc. of IEEE Conference on Decision and Control*, 2011.

[46] R. Ghaemi and D. Del Vecchio. Safety control of piece-wise continuous order preserving systems. In *Proc. of IEEE Conference on Decision and Control*, 2011.

[47] C. Le Guernic. Reachability analysis of hybrid systems with linear continuous dynamics. In *PhD thesis, Univerite Joseph Fourier*, 2009.

[48] J. A. Haddon, D. N. Godbole, A. Deshpande, and J. Lygeros. Verification of hybrid systems: Monotonicity in the AHS control system. In *Hybrid Systems III.* Lecture Notes in Computer Science, vol. 1066. Springer, 1996.

[49] M. R. Hafner and D. Del Vecchio. Computation of safety control for uncertain piecewise continuous systems on a partial order. In *Conf. on Decision and Control*, pages 1671 – 1677, 2009.

[50] M. R. Hafner and D. Del Vecchio. Computational tools for the safety control of a class of piecewise continuous systems with imperfect information on a partial order. *SIAM Journal of Control and Optimization*, 49:2463–2493, 2011.

[51] Michael R. Hafner and Domitilla Del Vecchio. Computational tools for the safety control of a class of piecewise continuous systems with imperfect information on a partial order. *SIAM J. Control and Optimization*, 49(6):2463–2493, 2011.

[52] J. K. Hedrick, Y. Chen, and S. Mahal. Optimized vehicle control/communication interaction in an automated highway system. *California Partners for Advanced Transit and Highways (PATH). Research Reports: Paper UCB-ITS-PRR-2001-29.* `http://repositories.cdlib.org/its/path/reports/UCB-ITS-PRR-2001-29`, 2001.

[53] T. A. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science*, pages 278–292. IEEE press, 1996.

[54] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. A user guide to HyTech. In *TACAS 95: Tools and Algorithms for the construction and analysis of systems,* Lecture Notes in Computer Science, vol. 1019, E. Brinksma, W. Cleaveland, K. Larsen, T. Margaria, and B. Steffen (Eds.), Springer-Verlag, pages 41–71, 1995.

[55] T. A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi. Beyond HyTech: Hybrid systems analysis using interval numerical methods. In *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, vol. 1790, B. Krogh and N. Lynch (Eds.), Springer Verlag, pages 130–144, 2000.

[56] T. A. Henzinger and Peter W. Kopke. Discrete-time control for rectangular hybrid automata. *Theoretical Computer Science*, 221:369–392, 1999.

[57] M. Heymann, F. Lin, and G. Meyer. Synthesis and viability of minimal interventive legal controllers for hybrid systems. *Discrete Event Dynamic Systems: Theory and Applications*, 8(2):105–135, 1998.

[58] R. J. Hill. Electric railway traction tutorial, part 1: Electric traction and dc traction motor drives. *Power Engineering Journal*, (1):47–56, 1994.

[59] R. Horowitz and P. Varaiya. Control design of an automated highway system. *Proceedings of the IEEE*, 88(7):913–925, Jul 2000.

[60] R. Horowitz and P. Varaiya. Control design of an automated highway system. *Proceedings of the IEEE*, 88(7):913–925, 2000.

[61] J. Hu, M. Prandini, and S. Sastry. Aircraft conflict prediction in the presence of a spatially correlated wind field. *IEEE Transactions on Intelligent Transportation Systems*, (3):326–340, 2005.

[62] H. Kowshik, D. Caveney, and P. R. Kumar. Provable systemwide safety in intelligent intersections. *IEEE Transactions on Vehicular Technology*, pages 804 – 818, 2011.

[63] A. B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for hybrid dynamics: the reachability problem. In *New Directions and Applications in Control Theory,* Lecture Notes in Control and Information Sciences, vol 321, W.P. Dayawansa, A. Lindquist, and Y. Zhou (Eds.), pages 193–205, 2005.

[64] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[65] T. Lawson. *Topology: A Geometric Approach*. Oxford, 2003.

[66] J. Lee and B. Park. Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment. *IEEE Trans. on Intelligent Transportation Systems*, 13(1):81–90, 2012.

[67] D. Liberzon. *Switching in Systems and Control*. Systems and Control: Foundations and Applications. Birkhäuser, Boston, 2003.

[68] J. Lygeros. An overview of hybrid systems control. *Handbook of Networked and Embedded Control Systems,* D. Hristu-Varsakelis and W. S. Levine (Eds.), Birkhäuser, pages 519–537, 2005.

[69] J. Lygeros, D. N. Godbole, and S. Sastry. A verified hybrid controller for automated vehicles. In *Conf. on Decision and Control*, pages 2289–2294, Kobe, Japan, 1996.

[70] J. Lygeros and N. Lynch. Strings of vehicles: Modeling and safety conditions. pages 273–288, 1998.

[71] J. Lygeros, C. J. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[72] V. Milanés, J. Pérez, E. Onieva, and C. González. Controller for urban intersections based on wireless communications and fuzzy logic. *IEEE Trans. on Intelligent Transportation Systems*, 11(1):243–248, 2010.

[73] G. K. Mitropoulos, I. S. Karanasiou, A. Hinsberger, F. Aguado-Agelet, H. Wieker, H-J Hilt, S. Mammar, and G. Noecker. Wireless local danger warning: Cooperative foresighted driving using intervehicle communication. *IEEE Trans. on Intelligent Transportation Systems*, 11(3):539–553, 2010.

[74] J. R. Munkres. *Topology*. Prentice Hall, second edition, 2000.

[75] U.S. DOT National Highway Traffic Administration (NHTSA). Traffic safety facts. 2003.

[76] U.S. DOT National Highway Traffic Administration (NHTSA). Vehicle Safety Communications Applications vsc-a, second annual report, 2008.

[77] M. Oishi, I. Mitchell, A. Bayen, and C. Tomlin. Invariance-preserving abstractions of hybrid systems: Application to user interface design. *IEEE Transactions on Control Systems Technology*, 16:229244, 2008.

[78] A. V. Oppenheim, A. S. Wilsky, and S. Hamid. *Signals and Systems*. Prentice Hall, second edition, 1996.

[79] J. Pachl. *Railway operation and control*. VTD Rail Publishing, 2002.

[80] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, vol. 2993, R. Alur and G. Pappas (Eds.), Springer Verlag, pages 477–492, 2004.

[81] A. Puri and P. Varaiya. Driving safely in smart cars. In *American Control Conference*, pages 3597–3599, Seattle, WA, 1995.

[82] R. Raffard, S. Waslander, A. Bayen, and C. Tomlin. A cooperative distributed approach to multi-agent eulerian network control: Application to air traffic management. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005.

[83] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, 1989.

[84] E. De Santis, M. D. Di Benedetto, and L. Berardi. Computation of maximal safe sets for switching systems. *IEEE Trans. Automatic Control*, 49(2):184–195, 2004.

[85] E. De Santis, M. D. Di Benedetto, and G. Pola. On observability and detectability of continuous-time linear switching systems. In *Conf. on Decision and Control*, pages 5777–5782, 2003.

[86] O. Shakernia, G. J. Pappas, and S. Sastry. Semi-decidable synthesis for triangular hybrid systems. In *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, vol. 2034, M. D. Di Benedetto and A. Sangiovanni-Vincentelli (Eds.), Springer Verlag, 2001.

[87] D. Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience, 2006.

[88] G. V. Smirnov. *Introduction to the Theory of Differential Inclusions*. American Mathematical Society, 2002.

[89] H. L. Smith. *Monotone Dynamical Systems*. Mathematical Surveys and Monographs. American Mathematical Society, 1995.

[90] W. Thomas. On the synthesis of strategies in infinite games. In *Proceedings of the STACS 95,* E. W. Mayr and C. Puech (Eds.), Springer Verlag, pages 1–13, 1995.

[91] C. J. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, 2000.

[92] C. J. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE*, 91(7):986–1001, 2003.

[93] U.S. DOT Turner-Fairbank Highway Research Administration (TFHRA). Roundabouts: An informational guide. http://www.tfhrc.gov/safety/00068.htm, 2000.

[94] P. Varaiya. Smart cars on smart roads. *IEEE Transactions on Automatic Control*, 38(2):195–207, Feb 1993.

[95] R. Verma and D. Del Vecchio. Safety control of hidden mode hybrid systems. *IEEE Trans. on Atuomatic Control*, 2011. To Appear.

[96] R. Verma and D. Del Vecchio. Safety control of hidden mode hybrid systems. *Automatic Control, IEEE Transactions on*, (99):1–1, 2012.

[97] R. Verma, D. Del Vecchio, and H. Fathy. Development of a scaled vehicle with longitudinal dynamics of a HMMWV for an its testbed. *IEEE/ASME Transactions on Mechatronics*, 13(1):46–57, 2008.

[98] R. Verma, D. Del Vecchio, and H. Fathy. Longitudinal vehicle dynamics scaling and implementation on a HIL setup. In *Proc. of the Dynamic Systems and Control Conference*, Ann Arbor, 2008. To appear.

[99] R. Verma and D. Del Vecchio. Continuous control of hybrid automata with imperfect mode information assuming separation between state estimation and control. In *Conf. on Decision and Control*, pages 3175 – 3181, 2009.

[100] R. Vidal, A. Chiuso, S. Soatto, and S. Sastry. Observability of linear hybrid systems. In *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, vol. 2623, O. Maler and A. Pnueli (Eds.), Springer Verlag, pages 526–539, 2003.

[101] H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *Conf. on Decision and Control*, pages 4607–4613, 1997.

[102] M. De Wulf, L. Doyen, and J.-F. Raskin. A lattice theory for solving games of imperfect information. *Hybrid Systems: Computation and Control,* Lecture Notes in Computer Science, vol. 3927, J. Hespanha and A. Tiwari (Eds.), Springer-Verlag, pages 153–168, 2006.