BARRIER FUNCTIONS AND MODEL FREE SAFETY WITH APPLICATIONS TO FIXED WING COLLISION AVOIDANCE

A Dissertation
Presented to
The Academic Faculty

By

Eric Squires

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2021

BARRIER FUNCTIONS AND MODEL FREE SAFETY WITH APPLICATIONS TO FIXED WING COLLISION AVOIDANCE

Approved by:

Dr. Magnus Egerstedt, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Samuel Coogan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Charles Pippin
Georgia Tech Research Institute
Georgia Institute of Technology

Dr. Yorai Wardi
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Zsolt Kira
School of Interactive Computing
Georgia Institute of Technology

Date Approved: July 28, 2021

To Veronica, Aubrey, and Emma. I love each of you dearly.

ACKNOWLEDGEMENTS

I would like to first thank my advisor, Dr. Magnus Egerstedt. You have provided a model of excellence that will continue to guide my research for years to come. I in particular appreciate how thoughtful you are in considering your students as you guide both their own research as well as encouraging fruitful collaboration. The chance to work with my other collaborators in GRITS, in particular Pietro Pierpaoli, Rohit Konda, and Samuel Coogan, has vastly improved not only this work but helped me improve as a researcher. To the rest of GRITS lab, I have thoroughly enjoyed my time in the lab. It has been a wonderful experience.

In addition to my time in the GRITS lab, I have been working at the Georgia Tech Research Institute where I have been thoroughly challenged and supported by my colleagues. The inspiration for a lot of this work has grown out of the need to solve practical issues that arise as we field high numbers of fixed wing aircraft that are collaborating in a small amount of space. To Charles Pippin, I appreciate the opportunities to investigate and grow in the Robotics Division the last few years, and your encouragement to continue on the path to getting a PhD. To Zsolt Kira, thanks for your encouragement to continue in deep learning. To my many other colleagues at GTRI, I have thoroughly enjoyed our collaboration and have been challenged to continually improve.

I can safely say this work would not have been completed without the steadfast encouragement of my wife Veronica. We discussed many times whether I should give up on this and you never allowed that to be the conclusion. Thanks for your unending encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Robotics is now being applied to a diversity of real-world applications [1] and in many areas such as industrial, medical, and mobile robotics, safety is a critical consideration for continued adoption. In this thesis we therefore investigate how to develop algorithms that improve the safety of autonomous systems using both a model-based and model-free framework. To begin, we make a variety of assumptions (e.g., that a model is known, there is a single safety constraint, there are no communication limits, and that the state can be sensed everywhere), and show how to guarantee the safety of the system. The contribution of the initial approach is a generalization of an existing method for creating a barrier function, which is a function similar to a Lyapunov function that can be used to make safety guarantees. We then investigate relaxing these initial assumptions. In some cases, new additional assumptions are required, performance may be reduced, or safety guarantees may no longer be available. We motivate the thesis with collision avoidance for fixed wing aircraft which can be viewed as a pairwise constraint on each pair of aircraft. This introduces the need for considering multiple safety factors simultaneously, and we show that an additional assumption is needed in this case. We then relax the assumption that the vehicles have unlimited communication and find that safety can still be guaranteed. However, it is possible in this case that the overriding safety controller may be more invasive than if more communication is allowed. When we then further relax the assumption that the state can be sensed at all times, safety can still be guaranteed in some specified situations but the system may be more permissive in approaching safety boundaries. We finally remove the assumption of a known model for dynamics. Although removing this assumption means the system is no longer guaranteed to be safe, the benefit is that it allows a safety designer to build a far less invasive override to get more performance out of the system.

CHAPTER 1
INTRODUCTION AND BACKGROUND

1.1 Introduction

Model-based engineering, where an environment or dynamics model is used to derive properties of a system, is responsible for a myriad of control systems, such as the Bernoulli equation that enables modern flight [2], to Lyapunov functions for humanoid robots [3], to barrier functions for safety [4]. These properties are often critical for real world operation as stability guarantees are important for predictable operations, derived convergence rates enable cost effective manufacturing plants, robustness results allow for robots to operate in a real world that consists of unmodelled effects, and safety guarantees are critical for systems interacting with humans.

Nevertheless, there can be limitations to model-based approaches. One example is a model mismatch where it may be necessary to assume a simplified model relative to the real system in order to derive properties. An example is where a unicycle model is assumed for fixed-wing aircraft collision avoidance [5, 6, 7] where closed form solutions are difficult to derive for more complicated 6 degree of freedom models. This can lead to differences between the predicted performance and actual performance on a live system. A second example of a limitation of a model-based approach can be performance. In computer vision, deep learning frameworks have risen in popularity as their performance has out-performed model-based methods [8]. Similarly, reinforcement learning, where an agent interacts with an environment and tries to maximize reward is another area where model free approaches can outperform model-based counterparts [9] and have led to human level performance in Atari [10], Starcraft [11], and Dota II [12].

Thus, there are tradeoffs to using model-based and model-free approaches. The former are more applicable when the system is well understood and the ability to predict system behavior is critical. The latter may be more applicable when the system is extremely complicated and final performance is important. Nevertheless, there have been a variety of approaches to combine these methods. For instance, in [9] the authors start with a model based controller and transition to a model free method at the end of training. The initial model allows the system to learn faster at the beginning but the final model free approach improves final performance. Other approaches have learned an environment model to then train an agent in reinforcement learning as this can improve robustness of the final result [13, 14]. Alternatively, in [15] there are two neural networks for a learned forward propagation model and model-free learning, respectively. A final action selection layer then uses these two networks.

Model based methods can also explicitly guide the trajectory of a learning agent. For instance, in [16], the authors learn the dynamics, value function, and safety model from a set of demonstration trajectories. They then apply model predictive control using these functions under the constraint that the planned path ends with sufficient probability in a known safe state. In [17] the authors use temporal logic to create an overriding controller that assigns a reward when there is a mismatch between the reinforcement learning and safety system. A similar idea is pursued in [18] where an overriding controller keeps the state in an admissible set and adds a cost for unsafe actions in order for the reinforcement learning system to also account for safety in its reward function.

For systems where safety is an important consideration, an override of a model free system can be pursued. For instance, in [19] a discrimination function is trained to mimic a human blocking action. Similarly, in [20] the authors train a reset policy to take a series of actions to get the agent back to safe initial conditions and is activated

when the reset policy value function gets below a threshold. Finally, in [21] it is shown that while blockers are effective at avoiding unsafe conditions, the use of a blocker can also be overly conservative, resulting in impaired overall performance.

This thesis investigates both model-based and model-free approaches to safety with a motivating example of fixed-wing collision avoidance, an application of increasing importance. As low-cost, unmanned aerial vehicles (UAVs) find civilian uses, the low-altitude airspace is increasingly congested, leading to large-scale UAV operation limitations including concerns for privacy, the environment, national security, and safe-flight validation [22]. A key challenge for safe-flight validation in congested environments is ensuring collision avoidance while enabling vehicles to accomplish their designed missions. Thus, in this thesis we propose an algorithm that minimally alters a vehicle's nominal control input while still ensuring safe operations.

A variety of approaches to fixed-wing collision avoidance have been proposed. Partially observable Markov decision processes are used in [23, 24] to achieve safe flight distances. Velocity obstacles [25] provide a geometric framework for selecting safe velocities. The dynamic window approach, originally introduced in [26] for static obstacles and adapted to moving obstacles in [27], uses circular arcs for trajectories and limits the set of allowable velocities to enable a quick optimization of the control input. In [5], the authors develop a first-order look-ahead algorithm that can be applied to vehicles with unicycle dynamics in a decentralized way while guaranteeing that collisions amongst k vehicles are avoided. Potential functions [28, 6] have also been applied to fixed-wing collision avoidance, where it can be shown that vehicles can safely avoid each other even when their sensing range is limited. Similarly, [7] discusses how to combine potential functions with trajectory goals into a navigation function in order to provide criteria under which collision avoidance can be guaranteed. Navigation functions have also been combined with Model Predictive Control (MPC) by making inter-agent distance requirements implicit in the cost function [29].

MPC has additionally been applied to UAV collision avoidance for vehicles with limited sensing [30] and communication constraints [31]. While MPC provides a flexible framework for distributed collision avoidance, its limited horizon can make safety guarantees difficult. In a more general case, the optimal control formulation in [32] allows for collision avoidance guarantees, but it is computationally intensive as it requires numerically solving the Hamilton-Jacobi-Bellman equations over an infinite horizon.

Trajectory generation was analyzed in [33] where a nonlinear program is developed to find a safe reference trajectory constructed from polynomials. In [34] and [35], the authors discuss trajectory generation using a randomly exploring random tree (RRT) with dynamics constraints provided by Dubins paths and a waypoint generation algorithm, respectively. Reference governors [36], where the input reference signal for a nominal closed loop controller is overridden in order to ensure that safety and performance constraints are maintained, have also been applied to collision avoidance in [37]. In [37] the authors show how to ensure collision avoidance for a distributed set of linear systems via a sequential mixed-integer programming optimization. The approach considers a finite horizon in the optimization because it is shown that a constant reference can then keep the system safe after that point. Reference governors are similar to the approach of this thesis in that given a nominal controller the approach seeks a minimal adjustment in order to improve safety characteristics. However, they differ in how the minimal adjustment occurs. A reference governor adjusts the set point that a nominal system is designed to achieve. On the other hand, the approach of this thesis does not require a reference input to the nominal system and instead allows a nominal controller to calculate a control input as it normally would. Finally, in [38], the authors also consider a trajectory based approach to avoid static obstacles. Similar to evasive maneuvers, traffic rules [39, 40] are a method for encoding hybrid behaviors that can include collision avoidance

trajectories. In [39], the authors show that a two vehicle system with limited sensing range can avoid collisions while reaching position goals. While in general this may result in conservative behaviors, they demonstrate in simulation that the decentralized algorithm continues to allow vehicles to reach their target configuration while avoiding collisions for as many as 70 vehicles. Reactive methods are useful because they can often be calculated online while evasive maneuvers benefit from a lookahead into the future. In this thesis we leverage the merits of both approaches within the framework of control barrier functions.

Motivated by the importance of formal guarantees of collision avoidance that are computationally feasible and minimally invasive we discuss in this thesis how to apply barrier functions (e.g., [41], [4]) to the UAV collision avoidance problem, where the system is subject to actuator constraints, nonlinear dynamics, and nonlinear safety constraints. Barrier functions are similar to Lyapunov functions and allow for guarantees that a system will stay safe (i.e., vehicles will maintain safe distances from each other) for all future times. Further, under some assumptions detailed later, a Quadratic Program (QP) can be used to calculate a safe control input implied by a barrier function so that the calculation can be done online [4]. Given such safety guarantees, barrier functions have been applied to a set of problems including collision avoidance for autonomous agents ([42, 43]), bipedal robots ([3, 44]), adaptive cruise control and lane following ([45, 4, 46, 47]), and in mobile communication networks [48].

However, barrier functions rely on being able to find a function with particular properties for safety to be guaranteed. For systems like a fixed wing UAV with actuator constraints, nonlinear dynamics, and nonlinear safety constraints, generating such a function can be difficult. In this respect they are similar to Lyapunov functions. They provide guarantees when a system designer can find appropriate functions but they may be difficult to construct.

Nevertheless, there are a variety of approaches to finding a barrier function given a system and safety constraints. One approach discussed for instance in ([45, 49, 41, 50]), uses a sum of squares decomposition [51]. In this approach an initially conservative estimate for a barrier function is found and the associated safe set is iteratively enlarged. Iterative approaches have also been developed when the system has relative degree greater than one. The conditions for calculating a safe control input for higher order systems are given in [52]. In [44], a backstepping approach is developed that ensures a control barrier function can be constructed and a similar approach is discussed in [53]. The approach discussed in this thesis is most similar to [54] where a barrier function is formulated by calculating the distance to a backup set after applying a backup controller. In this thesis we develop an alternative approach that does not require the specification of a backup set.

Geometric insights have also been exploited in [3], where the authors develop a barrier function for precise foot placement by ensuring that the foot is within the intersection of two circles. Similarly, in [42, 43], the authors develop a barrier function that ensures a circle and ellipsoid, respectively, around each robot will not overlap in order to ensure there will be no collisions for double integrator and quadrotor robots, respectively. Barrier functions have also been developed for unicycle dynamics in [47], where the dynamics are simplified by considering a point slightly in front of the vehicle.

Previous work on barrier functions has shown how, given the current state, a safe control input can be selected to ensure the system is safe for all times. In this paper, we also ensure system safety but do so by integrating the dynamics into the future using a known evasive maneuver that is always available to keep the system safe. In this respect the system is more predictable since it is known that a particular control input will be safe. Further, we ensure that actuator limits are respected which is a significant constraint in the case of UAVs where the system has non-zero minimum

velocity.

Aside from ensuring a barrier function constraint can be satisfied given actuator limitations, UAV collision avoidance also motivates the consideration of multiple safety constraints that must be satisfied at all times. In particular, because collision avoidance can be viewed as a constraint for each pairwise combination of vehicles [48, 55], we briefly review how barrier functions have been applied to systems with multiple constraints. A contract-based approach is presented in [45]. A sum of squares decomposition is presented in [50] where additional safety constraints map to additional constraints in the optimization problem. In [53], necessary and sufficient conditions are given for the existence of a control input that satisfies multiple barrier function constraints. The approach generalizes to high order and time-varying systems but requires that actuator constraints be unbounded. Barrier function composition has also been addressed in [45, 48, 55]. In [45], the authors partition the state space into regions for which a single barrier function is active in each component of the partition. In [48] and [55] non-smooth barrier functions are discussed, where the result allows for combining barrier functions using boolean primitives. One drawback of the boolean composition approaches is that it is not guaranteed that the composition of barrier functions will result in a barrier function.

Barrier functions have also been used in the context of limited sensing [42, 56, 57, 45] and allow for safety guarantees so that when the system starts safe it will remain safe for all future time. In [42] the authors provide a minimum sensing radius to ensure a system of double integrator robots maintain safe distances from each other. Further, they reformulate a Quadratic Program (QP) that only requires knowing the relative position to other agents while still ensuring safety for both collaborative and non-collaborative neighbors. In [56], the authors provide a decentralized strategy for collision avoidance that does not require knowing neighbor barrier function parameters. While [57] does not address multi-agent systems, it does consider collision

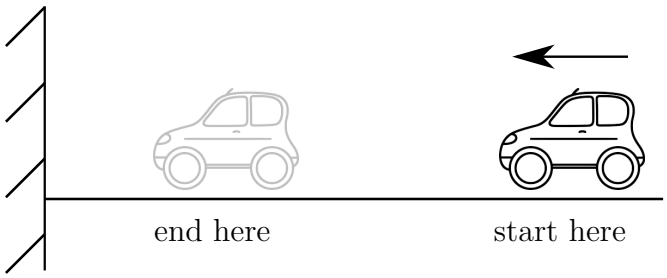


Figure 1.1: A simple system with a car that travels along the line. A safety override system should prevent the car from traveling towards the wall at high speed when it is close to the wall but can be more permissive when the car is far from the wall.

avoidance under limited range sensing for 3D quadrotors. The authors design a sequential QP that translates position-based constraints into rotational commands to ensure safety. Sensing limitations can also be addressed with a disturbance to the system dynamics. For instance, in [45] the authors model road curvature changes as a bounded disturbance to apply barrier functions to adaptive cruise control and lane keeping.

1.2 Barrier Functions Background

In this section we discuss how to generate a barrier function that ensures a system will stay safe for all future times. To motivate the discussion, consider the example in Figure 1.1 where a car has been designed to move along the x-axis in order to transport people. The car's controller applies acceleration commands to achieve speed set points and eventually slows down as it gets to the correct location. While it may seem better to design the car controller to achieve speed and safety goals simultaneously, this can significantly complicate controller design, particularly as dynamics, performance goals, or safety objectives become more complicated. For this reason, it may be that the car's controller may not be designed with safety goals in mind, like not hitting the wall shown in the figure.

Consider then how to design a safety system that, given the car's original controller, can allow the system to achieve its performance goal (get to a specific location on the x-axis) while not sacrificing the safety goal (do not hit the wall). Clearly, if the car is very close to the wall, the safety system should only allow the car to remain in place or move to the right. On the other hand, if the car is extremely far from the wall, the safety override can be very permissive because even if the car is moving very quickly towards the wall, there is still a substantial amount of time to slow down and avoid a collision. In between these two extremes, the safety override might try to balance allowing the car to approach the wall somewhat but not very quickly. In other words, it is not just how close to the wall the car is but how quickly it is approaching it that matters when considering how invasive a safety override should be.

A barrier function is a function that allows the system's safety designer to make this type of tradeoff precise. In particular, it allows us to calculate both how far the system is from the safety boundary as well as how quickly the system is approaching it. This facilitates a calculation of how much each of the available control inputs will affect how quickly the system approaches the boundary of safety. As a result, we can then pick a control value that is as close as possible to the original system's controller that does not violate the safety objective. We will refer to the set of available control inputs that do not approach the boundary too quickly as the admissible control space and the system's original controller as the nominal controller. Because a barrier function allows the safety designer to select a control value as close as possible to the nominal controller, we refer to the overriding safety controller as a minimally invasive controller. Because a barrier function allows the safety designer to select a control value as close as possible to the nominal controller, we refer to the overriding safety controller as a minimally invasive controller. We make this idea precise below.

Consider now a more realistic example of a fixed-wing aircraft which will have

more complicated dynamics than the simple car example of Figure 1.1. A common use case is for a remotely controlled UAV that is equipped with an autopilot that achieves waypoint commands. While the autopilot may be able to achieve waypoints, it may not factor other aircraft into the calculation of the control inputs. For vehicles operating in a dense region of space, simply using a waypoint following algorithm will lead to high risk of aircraft collisions which can significantly disrupt operations, lead to loss of aircraft, and place infrastructure below the crash at risk.

A simple solution to ensuring vehicles do not crash into each other is to be conservative in what the vehicles are allowed to do. However, this can lead to performance degradation. For the car example of Figure 1.1, this might mean that the car gets to its goal location more slowly or simply cannot get to locations near to the wall. Similarly, if high density operation is required for fixed-wing aircraft then an overly conservative approach to safety might mean the system's performance goal is compromised. However, as we discuss in this section, a barrier function allows the safety system to select an overriding control value that is minimally invasive so it can get as much performance out of the originally designed system without compromising safety.

To design a barrier function, one needs to specify a model and then develop a function that has particular properties that allow us to calculate both how far from the safety boundary the state is and how quickly it is approaching the boundary. We begin by specifying a model for fixed-wing aircraft. In particular, we assume small bank and pitch angles and note that similar models have been used in prior work for fixed-wing collision avoidance [5, 6, 7, 29, 39, 40]. We index each vehicle by i where $i \in \{1, \ldots, k\}$ where k is the total number of vehicles. The state of each vehicle i is then $x_i = \begin{bmatrix} p_{i,x} & p_{i,y} & \theta_i & p_{i,z} \end{bmatrix}^T$, where $p_{i,x}$, $p_{i,y}$, and $p_{i,z}$ are the x, y, and z positions of vehicle, respectively, and θ_i is the rotation of the vehicle. The control input for each vehicle is $u_i = \begin{bmatrix} v_i & \omega_i & \zeta_i \end{bmatrix}^T$, where v_i, ω_i, and ζ_i are the forward speed, rotation

speed, and vertical speed of the vehicle. We also assume that each vehicle can sense the state of every other vehicle so that the sensing problem is a complete undirected graph.

A critical consideration of this work is that the control inputs of the vehicle are bounded and in particular, that the aircraft must maintain positive forward speed at all times. The bounds for the control inputs are then $v_i \in [v_{min}, v_{max}]$ with $v_{min} > 0$, $|\omega_i| \leq \omega_{max}$, and $|\zeta_i| \leq \zeta_{max}$. Then the individual vehicle dynamics are described by

$$\dot{x}_i = \begin{bmatrix} \cos(\theta_i) & 0 & 0 \\ \sin(\theta_i) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \\ \zeta_i \end{bmatrix}. \tag{1.1}$$

Note that (1.1) is linear in the control input. Collision avoidance involves more than a single aircraft, so for this reason we consider the overall system state as the concatenation of all aircraft states

$$x = \begin{bmatrix} x_1 & x_2 & \dots & x_k \end{bmatrix}^T. \tag{1.2}$$

The system (1.2) is an instantiation of the more general system

$$\dot{x} = f(x) + g(x)u, \tag{1.3}$$

where f and g are locally Lipschitz functions, $x \in \mathbb{R}^n$, and $u \in U \subset \mathbb{R}^m$. Note that (1.3) is linear in the control input. This allows us to calculate an overriding control value using a Quadratic Program (QP) and will be described later. We also assume that solutions are forward complete, meaning the system has a unique solution for all time $t \geq 0$ given a starting condition $x(0)$. This assumption means that a model is well defined for all future times and is necessary in order to make claims that the

system will stay safe over an infinite horizon.

Given the state x, we can calculate properties of the state with an output function h. The discussion of the car example from Figure 1.1 emphasized that it is important to know both how far the system is from the safety boundary as well as how quickly it is approaching the safety boundary. The output function h allows us to calculate both values. To do this, we specify a set as a superlevel set of h, namely

$$\mathcal{C} = \{x \in \mathcal{D} \, : \, h(x) \geq 0\}. \tag{1.4}$$

In other words, when $h(x) = 0$, the system is at the boundary of safety and as $h(x)$ increases the system is further from the safety boundary. Further, the derivative of h measures how quickly the state is approaching the safety boundary. In the below definition $L_f h(x) = \frac{\partial h(x)}{\partial x} f(x)$ and $L_g h(x) = \frac{\partial h(x)}{\partial x} g(x)$ denote the Lie derivatives.

Definition 1. [4] Given a set $\mathcal{C} \subset \mathbb{R}^n$ defined in (1.4) for a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$, the function h is called a zeroing control barrier function (ZCBF) defined on an open set $\mathcal{D}$ with $\mathcal{C} \subset \mathcal{D} \subset \mathbb{R}^n$, if there exists a Lipschitz continuous extended class $\mathcal{K}$ function α such that

$$\sup_{u \in U} [L_f h(x) + L_g h(x) u + \alpha(h(x))] \geq 0, \; \forall x \in \mathcal{D}. \tag{1.5}$$

The terms $L_f h(x) + L_g h(x) u$ of (1.5) represents the time derivative of h when using u as a control input. In other words, for a fixed control input, (1.5) becomes

$$\dot{h}(x) \geq -\alpha(h(x)). \tag{1.6}$$

We can use (1.6) to show that the above definition encapsulates the idea that the state can approach the safety boundary quickly when it is far from the safety boundary.

First, note that for $h(x) = 0$, the derivative of h must be greater than or equal to zero. On the other hand, if $h(x) > 0$ then the derivative of h can also be negative. Further, the derivative of h can be increasingly negative the larger $h(x)$ becomes.

From Definition 1, it follows that the admissible control space is defined in [4] as

$$K(x) = \{u \in U \ : \ L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0\}. \tag{1.7}$$

This is the set of control inputs that ensure that the system is not approaching the safety boundary too quickly.

Theorem 1. [4] Given a set $\mathcal{C} \subseteq \mathbb{R}^n$ defined in (1.4) for a continuously differentiable function h, if h is a ZCBF on $\mathcal{D}$, then any Lipschitz continuous controller $u : \mathcal{D} \to U$ such that $u(x) \in K(x)$ will render the set $\mathcal{C}$ forward invariant.

Theorem 1 says that if we pick a control input so that the system does not approach the safety boundary too quickly then the system will stay safe for all future times. Importantly, any selection such that $u \in K(x)$ (provided it is Lipschitz continuous) will suffice to ensure safety. As discussed in [4], this means that when $K(x)$ has more than one element, we can introduce performance criteria to select the best one.

Consider again the example of an autopilot that is designed to reach a waypoint. We refer to the control input that is designed to reach a waypoint as the nominal controller and denote this function by $\hat{u}$. Suppose further that other aircraft are very far away so that $\hat{u}(x)$ is safe in the sense that it will not cause a collision. In particular assume that $\hat{u} \in K(x)$. Then by selecting the nominal control input, the system will not only stay safe but also achieve its originally intended performance goal.

However, for the case that the nominal control input is not safe (i.e., $\hat{u}(x) \notin K(x)$) we can still select a safe control input that is close to the nominal control input. For instance, suppose the nominal controller encodes a trajectory that points straight at a waypoint but that this trajectory will lead to a collision. We may still allow

the vehicle to deviate slightly from its intended target while still making progress by selecting a control value in $K(x)$ that is as close as possible to $\hat{u}$ as possible. Thus, choosing $u(x) \in K(x)$ ensures safety but choosing $u(x)$ as close as possible to $\hat{u}$ can retain performance, in the sense of being as close to the originally designed control value as possible.

This idea is formalized in [4], where a QP is introduced to choose how to calculate $u(x) \in K(x)$ while being as close as possible to the nominal control value as possible. A QP is an optimization problem where all of the constraints are linear and the optimization cost is quadratic. Notice in particular that the constraint in (1.5) is linear in u. Further, the constraints on the control input for fixed wing aircraft in (1.1) can be described by a set of linear inequalities as follows:

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \\ \zeta_i. \end{bmatrix} \geq \begin{bmatrix} v_{min} \\ -v_{max} \\ -\omega_{max} \\ -\omega_{max} \\ -\zeta_{max} \\ -\zeta_{max} \end{bmatrix}. \tag{1.8}$$

We denote the constraint (1.8) by $A_i u_i \geq b_i$ and note that for the system of k aircraft, the constraint becomes

$$Au = \begin{bmatrix} A_1 & 0 & 0 & \cdots & 0 \\ 0 & A_2 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & A_k \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_k \end{bmatrix} \geq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_k \end{bmatrix} = b. \tag{1.9}$$

Thus, the safety constraint (1.5) and actuator constraints (1.9) are both linear in the

control in put u.

In addition to linear constraints, the QP must have a quadratic cost. Thus, to encode the goal of picking a control input as close as possible to the nominal input $\hat{u}$ we can minimize $\frac{1}{2}\left\|u \hat{-} \hat{u}\right\|^2$ as follows

$$u^* = \min_{u \in \mathbb{R}^m} \frac{1}{2} \|u - \hat{u}\|^2 \tag{1.10a}$$

$$\text{s.t.} \quad L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0 \tag{1.10b}$$

$$Au \geq b. \tag{1.10c}$$

The QP in (1.10) resolves two practical issues with applying barrier functions. First, QPs can be solved very quickly so they are amenable to being run on UAVs where a safe control input may need to be calculated many times per second. Second, the QP will always be feasible. This is ensured because when h is a ZCBF as defined in Definition 1, it satisfies the constraints of the problem.

Thus, when there is a ZCBF h available, safety can be guaranteed and a control input can be calculated in an online manner. However, these strong conclusions are predictated on having a ZCBF available for the system. This is non-trivial for systems such as fixed wing aircraft due to actuator constraints and nonlinear dynamics. Nevertheless, we show in the next chapter how to derive such a ZCBF to guarantee safety for fixed wing UAV collision avoidance.

CHAPTER 2

CONSTRUCTING BARRIER FUNCTIONS

The last chapter introduced ZCBFs and how they can be used to make safety guarantees. However, it can be diffult to derive a ZCBF for a given system. We therefore begin this chapter with a motivating example for fixed wing collision where the candidate ZCBF does not satisfy the constraint (1.5) before presenting a general approach to deriving a ZCBF for the system.

2.1 Motivating Example

Consider a candidate ZCBF, h, that encodes a collision avoidance safety constraint in a system of two vehicles with state $x = \begin{bmatrix} x_1^T & x_2^T \end{bmatrix}^T$ and

$$h(x) = d_{1,2}(x) - D_s^2, \tag{2.1}$$

where

$$d_{1,2}(x) = (p_{1,x} - p_{2,x})^2 + (p_{1,y} - p_{2,y})^2 + (p_{1,z} - p_{2,z})^2$$

is the squared distance between vehicles 1 and 2 and D_s is a positive minimum safety distance. In other words, the candidate barrier function encodes the safety constraint directly. However, as we will show below, it does not fully account for the fact that vehicles need to start avoiding each other well in advance of a collision. Intuitively, we should not need wait for metal to crunch other metal to identify that the system is unsafe. Instead, we should know well in advance of the collision point that a collision is imminent which this candidate function does not encapsulate. We now examine this point mathematically.

To show why h defined in (2.1) is not a ZCBF, we present an example where,

even though the configuration of the aircraft is safe since $x \in \mathcal{C}$, $h(x)$ does not satisfy constraint (1.5). Let $x_1 = \begin{bmatrix} -D_s/2 & 0 & 0 & \epsilon \end{bmatrix}^T$ and $x_2 = \begin{bmatrix} D_s/2 & 0 & \pi & -\epsilon \end{bmatrix}^T$ for some $\epsilon \geq 0$. First, we note that for $x = \begin{bmatrix} x_1^T & x_2^T \end{bmatrix}^T \in \mathcal{C}$, $h(x) \geq 0$. Further,

$$\begin{aligned}
\sup_{u \in U}[L_f h(x) + L_g h(x) u + \alpha(h(x))] &= \sup_{u \in U} \big[2(p_{1,x} - p_{2,x})(v_1 \cos\theta_1 - v_2 \cos\theta_2) \\
&\quad + 2(p_{1,y} - p_{2,y})(v_1 \sin\theta_1 - v_2 \sin\theta_2) \\
&\quad + 2(p_{1,z} - p_{2,z})(\zeta_1 - \zeta_2)\big] \\
&= \sup_{u \in U}[-2D_s(v_1 + v_2) + 2\epsilon(\zeta_1 - \zeta_2)] \\
&= -4D_s v_{min} + 2\epsilon\zeta_{max}.
\end{aligned}$$

Since $v_{min} > 0$ and $D_s > 0$, if the two vehicles' initial positions satisfy $0 \leq \epsilon < 2D_s v_{min}/\zeta_{max}$ we observe that the quantity above does not satisfy constraint (1.5), i.e., $\sup_{u \in U}[L_f h(x) + L_g h(x) u + \alpha(h(x))] < 0$. Therefore, we conclude that $h(x)$ defined in (2.1) is not a ZCBF. The problem with this candidate ZCBF is that it does not account for the fact that by the time the vehicles are close to colliding, it may be too late to avoid each other due to the limited turning radius and positive minimum velocity.

Going back to the example of Figure 1.1, let the position of the vehicle be given by p and let it be at the wall when $p = 0$. Although the safety objective is to ensure that $p \geq 0$ at all times, we cannot define $h(x) = p$ because the car needs to brake well in advance of the wall in order to slow down in time. In this case, when the car is at the wall, $h(x) = p = 0$ so h indicates that the system is safe. However, if the car has negative velocity there is nothing that can be done to prevent it from colliding with the wall. Hence, a barrier function must encode not only that the system is safe at the current moment, but also that the system is safe for future times as well. To ensure this is the case, we now investigate how to define a barrier function in such a way as to ensure that there is always a control input to keep the system safe whenever

$h(x) \geq 0$.

2.2 Constructing a Barrier Function via Evading Maneuvers

In order to overcome the difficulties demonstrated in the example of Section 2.1, we introduce a method to systematically construct a ZCBF from a safety constraint. Let $\rho : \mathcal{D} \to \mathbb{R}$ be a safety function that represents the safety objective we want to satisfy at all times so that $\rho(x) \geq 0$ indicates that the system is safe. In the example from Section 2.1 for vehicles i and j,

$$\rho(x) = d_{i,j}(x) - D_s^2. \tag{2.2}$$

Alternatively, for the car example of Section 2.1, $\rho(x) = p$. Second, let $\gamma : \mathcal{D} \to U$ be a nominal evading maneuver. Section 2.3.2 discusses specific examples of γ for the UAV collision avoidance problem. For the car example of Figure 1.1, an example nominal evading maneuver is to accelerate to the right. Assuming γ has been selected, let

$$h(x; \rho, \gamma) = \inf_{\tau \in [0,\infty)} \rho(\hat{x}(\tau)), \tag{2.3}$$

be a candidate ZCBF where $\hat{x}$ and $\dot{\hat{x}}$ are given by

$$\hat{x}(\tau) = x + \int_0^\tau \dot{\hat{x}}(\eta) d\eta, \tag{2.4}$$

$$\dot{\hat{x}}(\tau) = f(\hat{x}(\tau)) + g(\hat{x}(\tau))\gamma(\hat{x}(\tau)). \tag{2.5}$$

For ease of notation, we will omit the time dependencies whenever the time is clear from the context. We assume in this paper that the solution (2.4) is well defined and

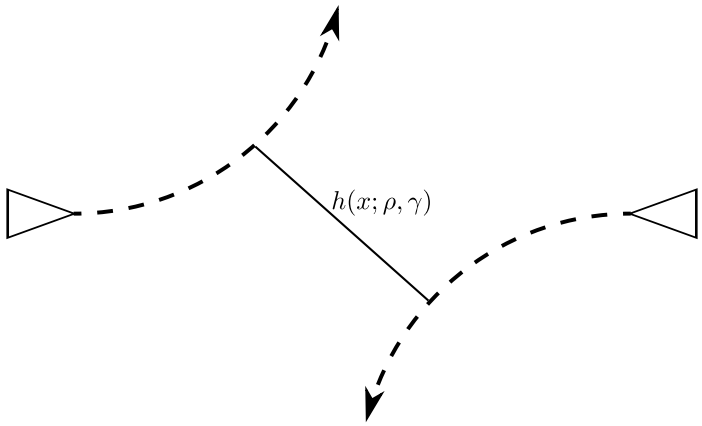


Figure 2.1: An example of how to calculate a barrier function for a system of two UAVs. In this case the vehicles start pointing at each other. The vehicles must maintain a minimum forward velocity so they cannot stop. Thus, they apply an evasive maneuver where both vehicles turn left. As the state is propagated using this evasive maneuver, the worse case distance along this trajectory is the value of the barrier function.

contained in $\mathcal{D}$ for all $\tau \geq 0$ so that $\rho(\hat{x}(\tau))$ is well defined. This choice of a candidate ZCBF h is motivated by the fact that in (2.3), h measures how close the state will get to the boundary of the safe set assuming γ is used as the control input for all future time. In other words, if the system is currently safe as defined by $h(x) \geq 0$ then by applying the nominal evading maneuver for all time the system will remain safe. An intuition behind the constructive method in (2.3) is shown in Figures 1.1 and 2.1.

In Section 2.1 we saw that we could not use the Euclidean distance for a ZCBF because when a candidate ZCBF h is defined as in (2.1), $K(x)$ could be empty for some $x \in \mathcal{D}$. In other words, although $x \in \mathcal{D}$ there was no control input available to keep the system safe. With h defined in (2.3), this problem is alleviated.

Theorem 2. Given a dynamical system (1.3) and a set $\mathcal{C} \subset \mathcal{D}$ defined in (1.4) for a continuously differentiable h defined in (2.3) with a safety function ρ and locally Lipschitz evading maneuver γ, h satisfies (1.5) for all $x \in \mathcal{C}$. If in addition, $L_g h(x)$ is non-zero for all $x \in \partial\mathcal{C}$ and γ maps to values in the interior of U, then h is a ZCBF on an open set $\mathcal{D}$ where $\mathcal{C} \subset \mathcal{D}$.

Proof. We start by assuming $x \in \mathcal{C}$ and show that h satisfies (1.5). Because $x \in \mathcal{C}$, $h(x) \geq 0$ so $\alpha(h(x)) \geq 0$. Further, note that $L_f h(x) + L_g h(x)\gamma(x)$ is the derivative along the trajectory of $\hat{x}$. In other words,

$$L_f h(x) + L_g h(x)\gamma(x) = \lim_{a \to 0^+} \frac{1}{a} \left(\inf_{\tau \in [a,\infty)} \rho(\hat{x}(\tau)) - \inf_{\tau \in [0,\infty)} \rho(\hat{x}(\tau)) \right). \tag{2.6}$$

Consider the term inside the parenthesis in (2.6), namely

$$\inf_{\tau \in [a,\infty)} \rho(\hat{x}(\tau)) - \inf_{\tau \in [0,\infty)} \rho(\hat{x}(\tau))$$

and notice that it is the subtraction of an infimum of the same function ρ evaluated on two different intervals. Further, note that the first interval is a subset of the second interval since a approaches 0 from above. Thus, the term inside the parenthesis on the right hand side of (2.6) is non-negative so $L_f h(x) + L_g h(x)\gamma(x) \geq 0$. We can then conclude that $L_f h(x) + L_g h(x)\gamma(x) + \alpha(h(x)) \geq 0$ so $\gamma(x) \in K(x)$.

Now assume that $L_g h(x)$ is non-zero for some $x \in \partial\mathcal{C}$ and γ maps to values in the interior of U. We will show that there is an open set $\mathcal{D}$ that is a strict superset of $\mathcal{C}$ for which (1.5) holds. Let $x \in \partial\mathcal{C}$ be such that $L_g h(x)$ is non-zero and $B(x, \mu)$ be a ball of radius $\mu > 0$ such that for all $z \in B(x, \mu) \setminus \mathcal{C}$, $L_g h(z)$ is non-zero. Such a ball exists such that $B(x, \mu) \setminus \mathcal{C}$ is nonempty because $L_g h(x)$ is continuous. Let $d(z)$ be a non-zero vector such that $d(z) + \gamma(x) \in U$ where $d(z)$ is a non-zero vector in the direction of $L_g h(z)$. Note that such a vector exists because γ maps to the interior of U. Also note that $L_g h(z)d(z) > 0$. Further restrict μ so that $L_g h(z)d(z) + \alpha(h(z)) \geq 0$ for all $z \in B(x; \mu) \setminus \mathcal{C}$. Note that for similar reasons discussed earlier in the proof, $L_f h(z) + L_g h(z)\gamma(z) \geq 0$. Then

$$L_f h(z) + L_g h(z)(\gamma(z) + d(z)) + \alpha(h(z)) \geq L_g h(x)d(z) + \alpha(h(z)) \geq 0.$$

□

Remark 1. The intuitive reason why h satisfies (1.5) whenever $h(x)$ is non-negative is that we have by definition a control input γ available to keep the system safe. A geometric view is presented in Figure 2.2. Note that γ is not the output of the Quadratic Program (1.10). Instead, the role of γ is to allow h to be evaluated via (2.3). In other words, there are three control values of interest in the safety override. First, the nominal controller, $\hat{u}$, is designed to accomplish a performance objective like achieving a waypoint. The goal is to find a safe control value as close as possible to $\hat{u}$. Second, the evasive maneuver γ exists only to calculate h. Its role is to ensure that the intersection of U and the hyperplane in Figure 2.2 is non-empty. Since this intersection can have more than one point, we use the optimization (1.10) to get as close as possible to $\hat{u}$ without violating the safety constraint. The solution to this optimization is the third control value of interest which is the actually applied safe control input u.

Remark 2. Theorem 2 holds for any class $\mathcal{K}$ function α. When $\alpha(h(x)) = 0$, (1.5) becomes $\dot{h}(x) \geq 0$. In other words, Theorem (2) can also be used to prove Lyapunov stability properties of a set by flipping the inequality.

2.3 Deriving a Barrier Function

2.3.1 Deriving a Barrier Function for Car Avoiding a Wall

We now discuss how to construct a barrier function for the example of Figure 1.1. To do so we first define the car model as a double integrator. The state is then $x = \begin{bmatrix} p & v_v \end{bmatrix}^T$ where p and v_v are the car position and velocity, respectively. The dynamics are given by

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} a_v$$

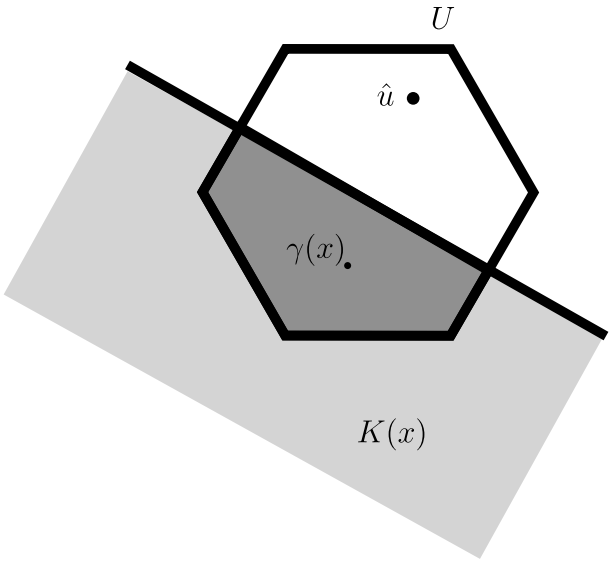


Figure 2.2: A geometric view of why h defined in (2.3) can be a barrier function. Here U is shown as a closed convex polytope satisfying $U = \{u : Au \geq b\}$ and $K(x)$ is the half-space. The constraint (1.5) implies that the intersection of U and $K(x)$ is non-empty. When h is defined in (2.3), it satisfies this constraint by ensuring that $\gamma(x) \in U$ and $\gamma(x) \in K(x)$ for all $x \in \mathcal{C}$.

where a_v is the acceleration of the vehicle which is subject to the constraint that $|a_v| \leq a_{v,max}$. The safety constraint in this case is that the position of the vehicle is nonnegative at all times. In other words, if the vehicle's position is negative then it has collided with the wall. For this reason we let $\rho(x) = p$. We then specify the evasive maneuver as a positive acceleration $a_{v,\gamma}$ that is less than $a_{v,max}$. We also denote the initial position and velocity as p_0 and $v_{v,0}$, respectively. Given this setup, the barrier function defined in (2.3) is given by

$$\begin{aligned} h(x) &= \inf_{\tau \in [0,\infty)} \rho(\hat{x}(\tau)) \\ &= \inf_{\tau \in [0,\infty)} p_0 + v_{v,0}\tau + \frac{1}{2} a_{v,\gamma} \tau^2. \end{aligned} \tag{2.7}$$

The τ that minimizes (2.7) is given by $\tau_{min} = \max(0, -v_{v,0}/a_{v,\gamma})$. Then (2.7) becomes

$$h(x) = p_0 + v_{v,0}\tau_{min} + \frac{1}{2} a_{v,max} \tau_{min}^2. \tag{2.8}$$

Note that (2.8) is continuously differentiable with respect to p and v_v for $\tau_{min} > 0$. For the case of $\tau_{min} = 0$, we verify that the derivative is the same whether $\tau_{min} = 0$ or $\tau_{min} = -v_{v,0}/a_{v,\gamma}$. In the first case where $\tau_{min} = 0$, $\frac{\partial h(x)}{\partial p} = 1$ and $\frac{\partial h(x)}{\partial v_v} = 0$. In the second case, where $\tau_{min} = -v_{v,0}/a_{v,\gamma}$ so that $v_{v,0} = 0$, we have $h(x) = p_0 - \frac{1}{2}v_{v,0}^2/a_{v,\gamma}$. Then $\frac{\partial h(x)}{\partial p} = 1$ and $\frac{\partial h(x)}{\partial v_v} = -v_{v,0}/a_{v,\gamma} = 0$ because $v_{v,0} = 0$.

2.3.2 Deriving a Barrier Function for UAV Collision Avoidance

We now consider how to calculate h defined in (2.3) for the UAV collision avoidance problem. From Theorem 2 the only restriction on γ and ρ is that γ is locally Lipschitz and that h is continuously differentiable so there is some flexibility in choosing γ and ρ. In this section we discuss two cases where we can choose γ and ρ so that h can be calculated in closed form. Let the initial state for vehicle i ($i = 1, 2$) be given by $\begin{bmatrix} p_{i,x_0} & p_{i,y_0} & \theta_{i,0} & p_{i,z_0} \end{bmatrix}^T$. For these examples we can calculate h in (2.3) for arbitrary initial states in closed form. Section 3 generalizes the results from Section 2.2 by showing how to calculate $k(k-1)/2$ barrier functions to ensure that the $k(k-1)/2$ pairwise distance constraints are always satisfied. Because the examples in this section calculate h in (2.3) using pairwise distance constraints, the calculations in these examples will also apply to the case of more than two vehicles. In other words, with the result of this section we can calculate barrier functions in closed form from arbitrary initial states and numbers of vehicles. Note that the solutions in this section solve for h in (2.3) in closed form where τ approaches infinity.

We emphasize that the specification of an evasive maneuver γ is necessary to evaluate h in (2.3). In other words, without a safety engineer specifying γ there cannot be a barrier function h. However, γ is never actually directly applied to the actuators. Instead, its role is to specify h so that the final actuator command u calculated in (1.10) can actually be applied to the aircraft. In this section we give two examples where for a given γ, h can be calculated in closed form even though it

is an integration over an infinite horizon. Further, while we provide two examples of an evasive maneuver to calculate a continuously differentiable h from (2.3) in closed form, we note that it a system specific derivation and we have not identified a general method for finding a γ for an arbitrary system that allows h to be calculated in closed form.

Example 1. In the first case, let

$$\gamma_{turn} = \begin{bmatrix} \sigma v & \omega & 0 & v & \omega & 0 \end{bmatrix}^T \tag{2.9}$$

with $0 < \sigma \leq 1$, $\omega \neq 0$. In other words, γ_{turn} is defined by the same turn rate for both vehicles but possibly different translational velocities. See Figure 2.1 for an example. Define $r = \frac{v}{\omega}$ to be the turn radius of the evasive maneuver when traveling at speed v, $b_{1,0} = p_{1,x_0} - \sigma r \sin(\theta_{1,0})$, $b_{2,0} = p_{2,x_0} - r \sin(\theta_{2,0})$, $c_{1,0} = p_{1,y_0} + \sigma r \cos(\theta_{1,0})$, $c_{2,0} = p_{2,y_0} + r \cos(\theta_{2,0})$, $\Delta b_0 = b_{1,0} - b_{2,0}$, $\Delta c_0 = c_{1,0} - c_{2,0}$, and $\delta > 0$. Let

$$\rho(x) = d_{1,2}(x) - 2\delta + \delta \sin(\theta_1) - \delta \cos(\theta_1) - D_s^2, \tag{2.10}$$

where the δ terms are introduced to affect the smoothness of h. See the Appendix for details. Note that given γ_{turn} the solution to the dynamics (1.1) for vehicles 1 and 2 are

$$x_1(\tau) = \begin{bmatrix} b_{1,0} + \sigma r \sin(\omega\tau + \theta_{1,0}) \\ c_{1,0} - \sigma r \cos(\omega\tau + \theta_{1,0}) \\ \omega\tau + \theta_{1,0} \\ p_{1,z_0} \end{bmatrix} \quad \text{and} \quad x_2(\tau) = \begin{bmatrix} b_{2,0} + r \sin(\omega\tau + \theta_{2,0}) \\ c_{2,0} - r \cos(\omega\tau + \theta_{2,0}) \\ \omega\tau + \theta_{2,0} \\ p_{2,z_0} \end{bmatrix},$$

respectively. Then

$$\begin{aligned} h(x) = \inf_{\tau\in[0,\infty)} & \left(\Delta b_0 + \sigma r\sin(\omega\tau+\theta_{1,0}) - r\sin(\omega\tau+\theta_{2,0})\right)^2 \\ & + \left(\Delta c_0 - \sigma r\cos(\omega\tau+\theta_{1,0}) + r\cos(\omega\tau+\theta_{2,0})\right)^2 \\ & + \left(p_{1,z_0} - p_{2,z_0}\right)^2 - 2\delta + \delta\sin(\omega\tau+\theta_{1,0}) - \delta\cos(\omega\tau+\theta_{1,0}) - D_s^2. \end{aligned}$$

By expanding the square terms and applying two trigonometric identities,[1] we get

$$\begin{aligned} h(x) = \inf_{\tau\in[0,\infty)} & \Delta b_0^2 + \Delta c_0^2 + (1+\sigma^2)r^2 - 2\sigma r^2\cos(\theta_{1,0}-\theta_{2,0}) \\ & + 2\sigma\Delta b_0 r\sin(\omega\tau+\theta_{1,0}) - 2\Delta b_0 r\sin(\omega\tau+\theta_{2,0}) \\ & - 2\sigma\Delta c_0 r\cos(\omega\tau+\theta_{1,0}) + 2\Delta c_0 r\cos(\omega\tau+\theta_{2,0}) \\ & + \left(p_{1,z_0} - p_{2,z_0}\right)^2 - 2\delta + \delta\sin(\omega\tau+\theta_{1,0}) - \delta\cos(\omega\tau+\theta_{1,0}) - D_s^2. \end{aligned} \tag{2.11}$$

Grouping constant terms and applying phasor addition yields

$$h(x) = \inf_{\tau\in[0,\infty)} A_1 + A_2\cos(\omega\tau+\Theta) - D_s^2, \tag{2.12}$$

where A_1 results from grouping constant terms, while A_2 and Θ are the amplitude and phase resulting from the phasor addition so that A_1 and A_2 are functions of x. By convention A_1 and A_2 are nonnegative with appropriate calculation of Θ. The minimum in (2.12) then occurs at $\tau = (\pi - \Theta + l2\pi)/\omega$ for integers l resulting in nonnegative t so that $h(x) = A_1 - A_2 - D_s^2$. Note that for the case where

$$\rho(x) = \sqrt{d_{1,2}(x) - 2\delta + \delta\sin(\theta_1) - \delta\cos(\theta_1)} - D_s, \tag{2.13}$$

[1]The identities are $\sin^2(\alpha)+\cos^2(\alpha) = 1$ and $\cos(\alpha-\beta) = \cos(\alpha)\cos(\beta) + \sin(\alpha)\sin(\beta)$.

the same reasoning yields

$$h(x) = \sqrt{A_1 - A_2} - D_s \tag{2.14}$$

for ρ defined in (2.13). We refer to the h in (2.14) as h_{turn}. To ensure that the square root is well defined, we must then require that $A_1 - A_2 \geq 0$ which occurs when the vehicles do not get more than 2δ from each other along the trajectory defined by (2.4) using γ_{turn} in (2.9). Since δ can be chosen to be arbitrarily small, it can be chosen so that $\delta \ll D_s$ so the vehicles are very far outside the safe set before this condition occurs.

Example 2. For a second case, let ρ be given in (2.2) and

$$\gamma_{straight} = \begin{bmatrix} v_1 & 0 & \zeta_1 & v_2 & 0 & \zeta_2 \end{bmatrix}^T, \tag{2.15}$$

where $v_1 \neq v_2$. An example of the barrier function constructed from this ρ and γ is shown in Figure 2.3. In other words, $\gamma_{straight}$ uses a zero turn rate while allowing the vehicles to have different speeds. In this case we have

$$\begin{aligned} h(x) = \inf_{\tau \in [0,\infty)} & \left(p_{1,x_0} + \tau v_1 \cos(\theta_{1,0}) - p_{2,x_0} - \tau v_2 \cos(\theta_{2,0})\right)^2 \\ & + \left(p_{1,y_0} + \tau v_1 \sin(\theta_{1,0}) - p_{2,y_0} - \tau v_2 \sin(\theta_{2,0})\right)^2 \\ & + \left(p_{1,z_0} + \tau \zeta_1 - p_{2,z_0} - \tau \zeta_2\right)^2 - D_s^2, \end{aligned} \tag{2.16}$$

which is quadratic in τ so the minimum can be calculated in closed form. See the Appendix for an analysis of the differentiability of h in this case. We refer to h constructed from $\rho(x) = \sqrt{d_{1,2}(x)} - D_s$ and $\gamma_{straight}$ as $h_{straight}$.

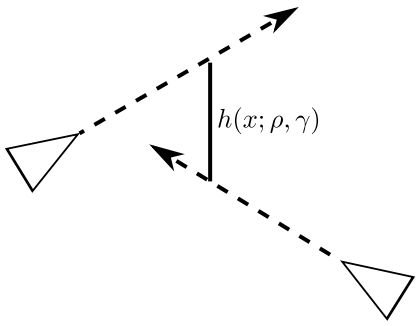


Figure 2.3: An example of a trajectory using an evasive maneuver where each vehicle maintains a straight trajectory.

2.4 Some Considerations In Choosing An Evasive Maneuver

In this section we consider design tradeoffs in selecting a nominal evasive maneuver. As the previous section suggests, a key aspect of choosing an evasive maneuver is being able to find in closed form the worst case future safety function value when using the evasive maneuver for all times. This was shown to be possible for both γ_{turn} and $\gamma_{straight}$ in Section 2.3.2. Given multiple evasive maneuvers, we now consider how the selection of an evasive maneuver can affect the observed performance the system.

We first discuss how the choice of an evading maneuver can affect the robustness of the system to noise in the dynamics. Note that Theorem 2 requires that an evasive maneuver be in the interior of the actuator constraints U. In particular, this assumption allows the system to satisfy (2.3) even when the state is outside of the safe set. Intuitively, the reason why it is important to define an evasive maneuver well within actuator limits is that it allows for the control input to adjust in a neighborhood around the evasive maneuver if anything goes wrong, as can happen when there is a perturbation in the dynamics or the system starts in a configuration such that $h(x) < 0$. The property that h satisfies (1.5) on an open set larger than $\mathcal{C}$ was shown to make the system robust to perturbations in the dynamics in [46] and to be important for ensuring system safety in Example 4 of [58]. Hence, it is important to

select an evasive maneuver that operates well within the actuator limits.

Robustness to perturbations in dynamics is important since any model will have some level of error in it when compared to a real world system. When the system is robust it can possibly allow for a simpler model to be used when designing a controller. We now consider a related idea, namely how a simple evasive maneuver might restrict the available set of final overriding control values. For instance, notice that the evasive maneuvers in (2.9) and (2.15) (when $\zeta_1 = \zeta_2 = 0$ in (2.15)) both encode trajectories where the vehicles maintain the same altitude for all times. This might indicate that the overriding safety controller might be limited to planar maneuvers and is therefore not exploiting an important evasive capability of the aircraft, namely the ability to change altitudes. However, this is not actually the case. Although γ_{turn} and $\gamma_{straight}$ (for $\zeta_1 = \zeta_2 = 0$) are purely planar maneuvers, they nevertheless can induce behaviors that exploit altitude changes. To see this, note that for h in (2.12) and (2.16),

$$\frac{\partial h(x)}{\partial p_{1,z_0}} = 2(p_{1,z_0} - p_{2,z_0}), \tag{2.17}$$

which is not equal to zero for $p_{1,z_0} \neq p_{2,z_0}$. A similar calculation also holds for $\frac{\partial h(x)}{\partial p_{2,z_0}}$. In other words, h changes as a function of initial altitude. Specifically, this means that the QP can exploit ζ_1 and ζ_2 because the fourth and eighth elements of $L_g h(x(t))$ are non-zero when $p_{1,z_0} \neq p_{2,z_0}$, i.e., the QP in (1.10) can exploit the altitude control input even though γ_{turn} and $\gamma_{straight}$ do not necessarily include an altitude changing term in the evasive maneuver. Similarly, although $\gamma_{straight}$ appears to not encode the ability to turn, we will observe in the simulation experiments of the next section that the overriding control value does allow the system to turn.

This confirms an important distinction between the distinct control values we consider in this thesis, namely the safe control value $u(x)$ and the evasive maneuver $\gamma(x)$. The evasive maneuver only exists to facilitate the calculation of the barrier

function h. Unlike the safe control value, it does not ever need to be applied to the system unless it happens to be a solution to the QP (1.10). However, the specification of the evading maneuver γ can have a significant effect on the safe control value u.

2.5 Simulation of Two Vehicles

We demonstrate the theoretical development of this section in simulation using SCRIMMAGE [59]. SCRIMMAGE is a multi-agent simulator designed to scale to high numbers of vehicles and includes a plugin-interface that makes it easy to experiment with different motion models and controllers without having to change code. This makes it simple to swap out nominal controllers and vary the fidelity of fixed-wing UAVs from the unicycle dynamics in (1.1) used in this section up to a 6-DOF model.

For the simulation, let k vehicles be positioned in a circle of radius 200 around the origin, where $k = 2$ in this simulation. In other words, vehicle i has initial state $x_i = \begin{bmatrix} 200\cos\left(i\frac{2\pi}{k} + \pi\right) & 200\sin\left(i\frac{2\pi}{k} + \pi\right) & i\frac{2\pi}{k} + \psi & \epsilon_i \end{bmatrix}^T$, where ψ is an additional offset so that vehicles are not necessarily starting with orientation pointing at the origin. The goal position for vehicle i is on the other side of the origin: $x_{i,g} = \begin{bmatrix} 200\cos\left(i\frac{2\pi}{k}\right) & 200\sin\left(i\frac{2\pi}{k}\pi\right) \end{bmatrix}^T$.

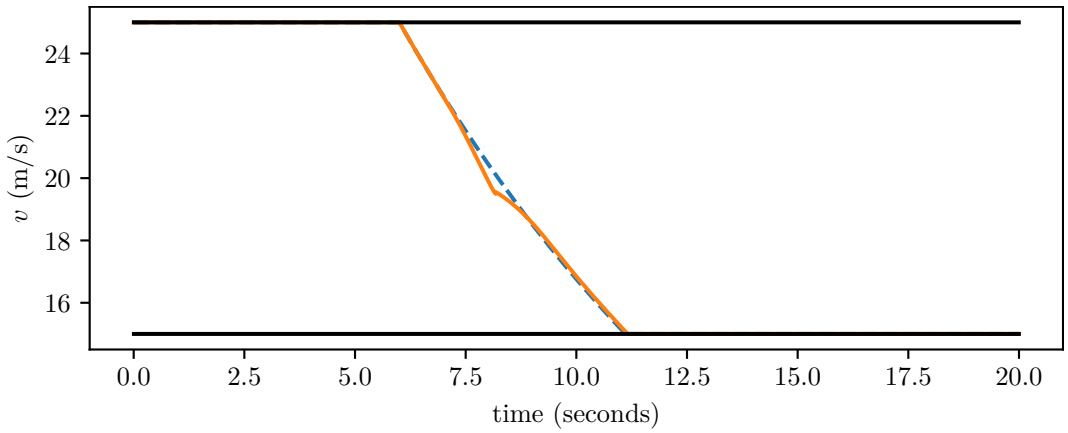
This setup is selected so that the vehicles are on a collision course. The nominal controller is that described in [60] with constant $\lambda = 1$. Additionally, we let $v_{min} = 15$ meters/second, $v_{max} = 25$ meters/second, $\zeta_{max} = 3.9$ meters/second, $\omega_{max} = 13$ degrees/second, $D_s = 5$ meters, and $\delta = 0.01$ meters^2. The choice of ζ_{max} results from assuming a maximum pitch of 15 degrees while traveling at v_{min}. ω_{max} is chosen to be consistent with a constant rate turn [61] with a 30 degree bank with a speed of v_{max}. We note that while the experiments do not consider dynamics or sensor noise, the robustness of barrier functions to noise was previously discussed in [46]. Each vehicle evaluates (1.10) at each timestep where we use OSQP [62] to evaluate the QP. We investigate the performance of the vehicles when h defined in (2.3) is constructed from

γ_{turn} in (2.9) and $\gamma_{straight}$ (2.15), respectively, where $\gamma_{turn} = \begin{bmatrix} v & \omega & 0 & v & \omega & 0 \end{bmatrix}^T$, $\gamma_{straight} = \begin{bmatrix} v & 0 & 0 & v & 0 & 0 \end{bmatrix}^T$, and $v = 0.9v_{min} + 0.1v_{max}$ and $\omega = 0.9\omega_{max}$. For the scenario with γ_{turn}, we let $\psi = 0$ so that the vehicles start with orientation pointing at the origin. For the scenario with $\gamma_{straight}$, we let $\psi = 2^\circ$ because if the vehicles pointed at the origin they would not start in the safe set. Additionally, for the γ_{turn} case we use ρ in (2.13). Similarly, for the $\gamma_{straight}$ case we use $\rho(x) = \sqrt{d_{1,2}(x)} - D_s$. Details of the distance between the vehicles and control signals are shown in Figures 2.4 and 2.5. Note that the resulting trajectory can be different depending on which γ is used as shown in Figure 2.5b. Nevertheless, in both cases the vehicles are able to maintain safe distances from each other and satisfy actuator constraints throughout the simulation regardless of which γ is used to construct a h.

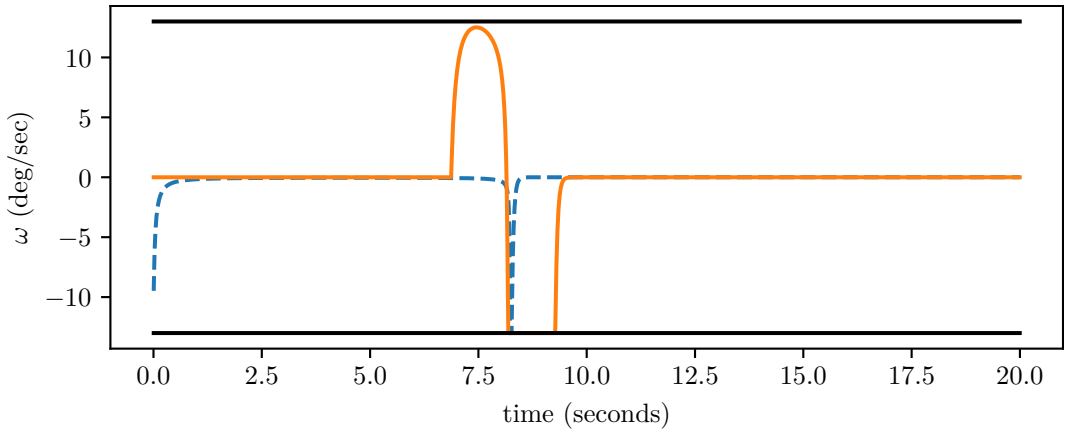
In the second experiment, we examine the effect of altitude control on the evasive behavior of the aircraft. Because (2.17) predicts that $\frac{\partial h(x)}{\partial p_{i,z_0}} \neq 0$ (for $i = 1, 2$) only when the vehicles are not at the same altitude, we start the vehicles at an altitude of -1 and 1, respectively. This offset is small enough to ensure that the nominal path of the vehicles still involves a collision. As was done in the previous experiment, we set $\psi = 0^\circ$ and $\psi = 2^\circ$ degrees when using γ_{turn} and $\gamma_{straight}$, respectively. In Figure 2.6 we show the output of ζ_1, where overriding behavior peaks around 8.2 seconds. Notice that the actuator output is within the limits of $\pm\zeta_{max}$. Further, the vehicles maintain safe distances at all times. This occurs even though the evading maneuver does not explicitly encode altitude changes.

2.6 Conclusion

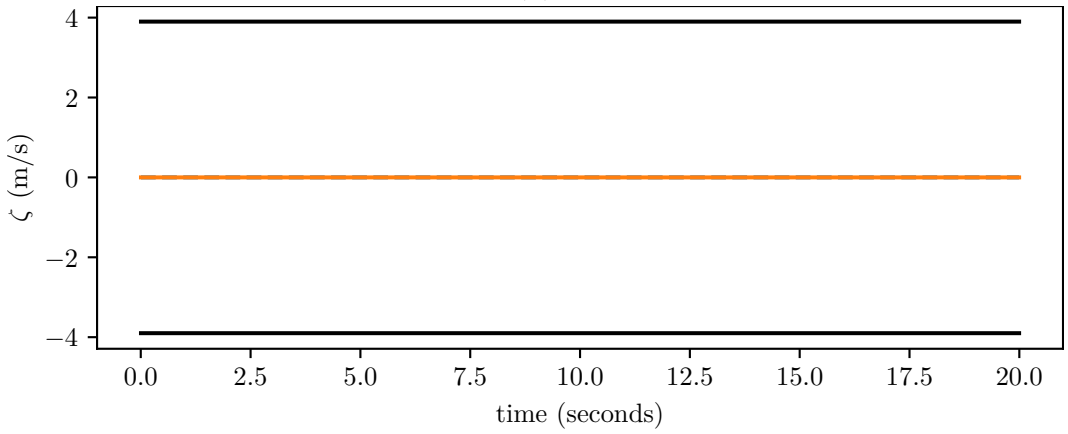
In this chapter we have shown how to ensure that two fixed wing aircraft do not collide. However, it involved a series of assumptions. In the rest of the thesis we examine whether similar conclusions can still be achieved when these assumptions



(a)

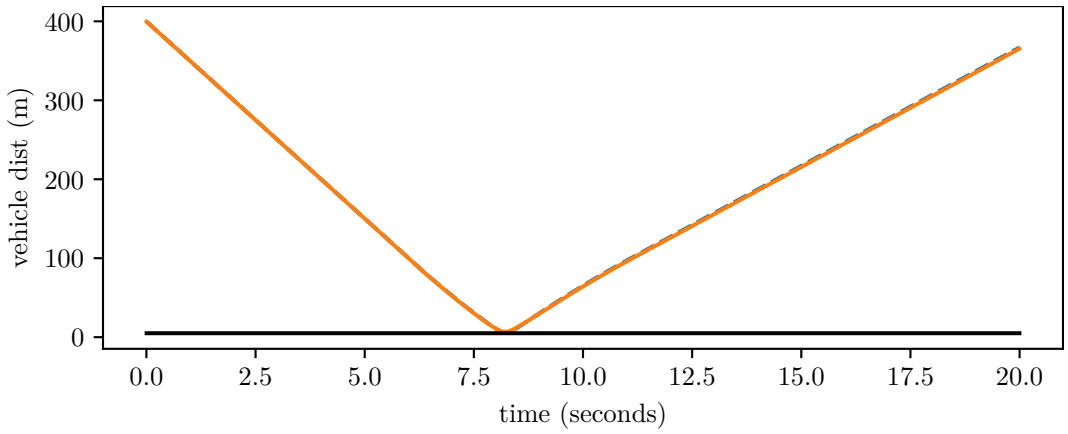


(b)

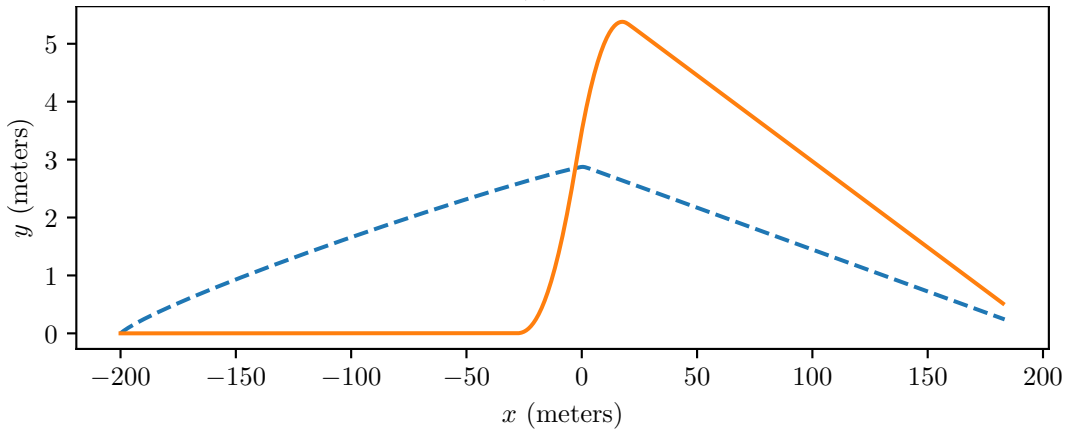


(c)

Figure 2.4: Control outputs for the scenario with 2 fixed-wing vehicles. The blue dashed and orange solid lines are the output of the scenario where h is constructed from $\gamma_{straight}$ and γ_{turn}, respectively. The barrier function chooses control values so that vehicle 1 velocity, turn rate, and altitude rates are within the actuator limits in (a), (b), and (c), respectively. Further note that the control values are within the actuator constraints. Adapted with permission from [63] ©2018 IEEE.

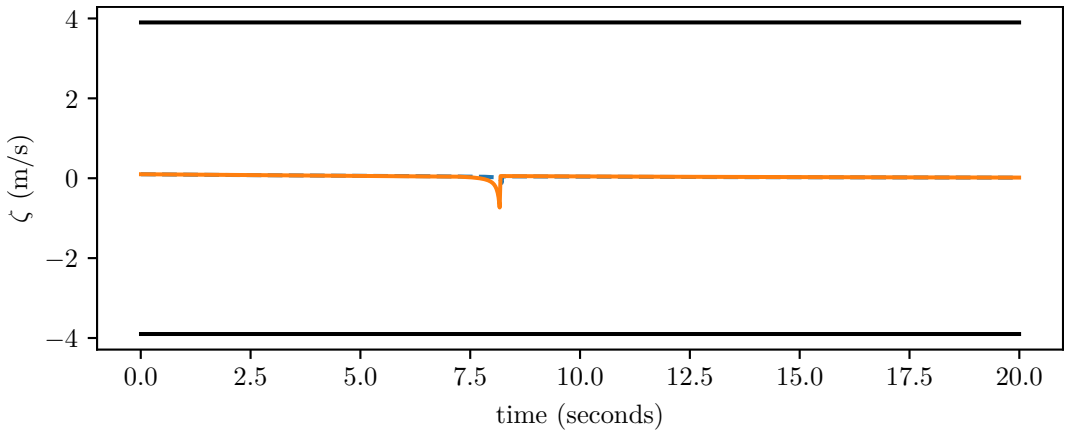


(a)

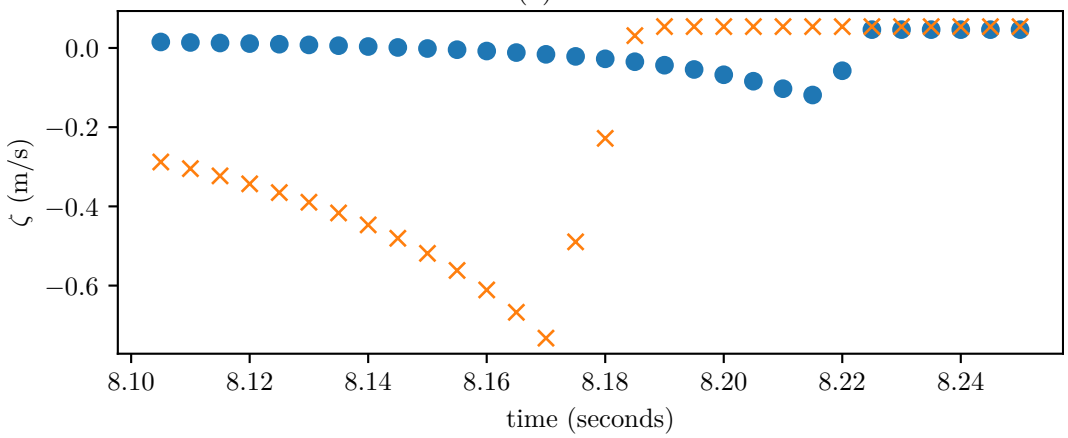


(b)

Figure 2.5: Safety outputs for the scenario with 2 fixed-wing vehicles. The blue dashed and orange solid lines are the output of the scenario where h is constructed from $\gamma_{straight}$ and γ_{turn}, respectively. The minimum distance between the vehicles is shown to be above D_s in (a) where the output is very similar in both scenarios. The path taken by vehicle 1 is shown in (b). Note that the choice of γ in constructing h has a significant effect on the path taken. Adapted with permission from [63] ©2018 IEEE.



(a)



(b)

Figure 2.6: (a) A plot of ζ_1 as a function of time when h is parameterized by γ_{turn} and $\gamma_{straight}$, respectively. Note the overriding control values around 8.2 seconds and that ζ_1 is within $\pm\zeta_{max}$. (b) The same plot zoomed in with individual control values plotted to indicate that the control signal does not experience abrupt changes.

Table 2.1: A list of initial assumptions required to ensure system safety

Assumption	Effect of Removing Assumption
One Safety Constraint	see Chapter 3
Unlimited Communication	see Chapter 4
Infinite Range Sensors	see Chapter 5
Known Dynamics Model	see Chapter 6

are relaxed. See Table 2.1 for a summary of these assumptions. In some cases, safety can still be guaranteed but there may be additional assumptions necessary or reduced performance. We find that the only assumption we cannot relax without giving up a strict safety guarantee is the assumption of a known model for the dynamics. Nevertheless, even in this case we are able to derive an algorithm where, even though there are non-zero observed collisions, the experimental collision rates are significantly reduced when using the proposed algorithm over the nominal system. Further, the algorithm that does not require dynamics allows the system designer to get more performance out of the system than the approach of this chapter.

CHAPTER 3

COMPOSITION OF MULTIPLE SAFETY CONSTRAINTS

The previous chapter showed how to create a framework to maximize the performance of a system while ensuring that one safety constraint is always satisfied. In this chapter we relax the assumption that only one safety constraint must be satisfied at all times and instead consider the case of satisfying multiple constraints simultaneously. By solving the problem of how to ensure safety while maximizing performance of the system under arbitrarily many safety constraints, we are able to solve the specific problem of arbitrarily many aircraft in a dense space without collisions.

We motivate the need for multiple safety constraints by introducing additional vehicles into the collision avoidance problem. This means that while the prior chapter only required that vehicles 1 and 2 maintain safe distances, we now must consider, in the case of three vehicles, how to ensure that vehicles 1 and 2 do not collide in addition to considering the distances between vehicles 1 and 3 as well as vehicles 2 and 3. We generalize the results to arbitrarily many vehicles.

A straightforward application of the previous chapter might be to create a barrier function for each safety constraint in the system (e.g., a ZCBF to ensure vehicles 1 and 2 cannot collide, another ZCBF to ensure vehicles 1 and 3 cannot collide, etc) and at each timestep have each vehicle select a safe actuator command satisfying all of the constraints. However, while the prior chapter showed that there exists some actuator value that satisfies each individual ZCBF constraint, it may be the case that there is not an actuator value to satisfy all ZCBF constraints simultaneously. To motivate the discussion, we begin by presenting one such example. The essential issue is that the safety override to ensure one constraint is safe may lead to the other constraint being violated and vice versa.

Having shown that having a set of individual barrier functions is not sufficient to guarantee that all safety constraints will be satisfied at all times, we introduce an additional assumption called the shared evading maneuver assumption. This assumption requires that the same evading maneuver is used to ensure safety for each constraint. Recall that in the previous chapter we demonstrated that constructing a barrier function with a nominal evading maneuver means the system can always use the evading maneuver to satisfy the safety constraint. Similarly, the shared evading maneuver assumption means that the evading maneuver can be used to satisfy each individual safety constraint, and as a consequence, it can be used to satisfy all constraints.

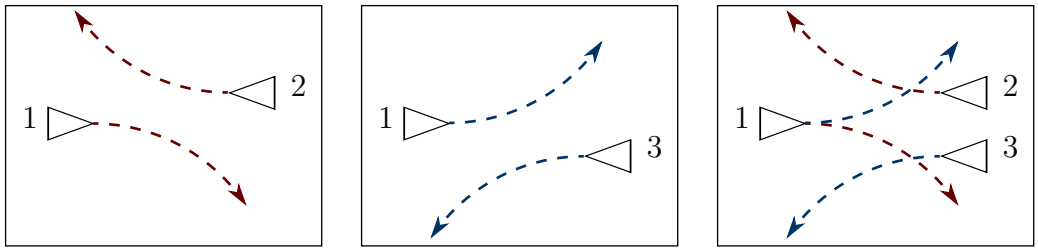
3.1 Motivating Example

Although the constructive method introduced in (2.3) can produce a barrier function in the presence of actuator constraints that ensures two vehicles do not collide, the formulation does not extend immediately to collision avoidance for systems with more than two vehicles. To see this, we present a specific example where three UAVs with a collision avoidance safety objective cannot use the results from Section 2.2 to ensure safety. A plot of this scenario is shown in Figure 3.1. We index the vehicles by $i = 1, 2, 3$.

To ensure collision-free trajectories, and considering the safety function defined in (2.10), three pairwise constraints must be nonnegative at all times:

$$\begin{aligned}
\rho^1(x) &= d_{1,2}(x) - 2\delta + \delta\sin(\theta_1) - \delta\cos(\theta_1) - D_s^2, \\
\rho^2(x) &= d_{1,3}(x) - 2\delta + \delta\sin(\theta_1) - \delta\cos(\theta_1) - D_s^2, \\
\rho^3(x) &= d_{2,3}(x) - 2\delta + \delta\sin(\theta_2) - \delta\cos(\theta_2) - D_s^2.
\end{aligned}$$

When these three functions are nonnegative for all future times, the vehicles will all



(a) (b) (c)

Figure 3.1: A geometric view of the example given in Section 3.1. In (a), h^1 defined in (2.3) is constructed to design a function so that vehicles 1 and 2 stay safe. Here γ^1 encodes an evasive maneuver where vehicles 1 and 2 turn right. Further, vehicles 1 and 2 are placed so that turning right is the only available control input to keep the system safe. In (b), a similar setup is shown for vehicles 1 and 3 where h^2 has been constructed from γ^2 which encodes an evasive maneuver where vehicles 1 and 3 turn left and vehicles 1 and 3 are placed so they are only able to turn left to stay safe. In (c), vehicle 1 cannot turn both right and left to avoid vehicles 2 and 3, respectively. Although vehicle 1 can avoid them individually, it cannot avoid them both simultaneously.

maintain safe distances from each other. We now apply the results of Section 2 to these constraints and for simplicity, let δ be approximately 0. For each constraint, define an arbitrarily chosen nominal evading maneuver

$$\gamma^1(x) = \begin{bmatrix} 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 \end{bmatrix}^T \tag{3.0a}$$

$$\gamma^2(x) = \gamma^3(x) = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}^T. \tag{3.0b}$$

In other words, γ^1 encodes an evasive maneuver where all the vehicles turn right while γ^2 and γ^3 encode a maneuver where all the vehicles turn left. We note that h^j ($j = 1, \ldots, 3$) defined in (2.3) and constructed from ρ^j and γ^j are ZCBFs. In other words, safety can be guaranteed for a single safety constraint (e.g., that vehicles 1 and 2 will not collide) but not necessarily all three constraints (e.g., that vehicles 1 and 3, or 2 and 3, will not collide). In this example we let $v_{min} = 1$, $v_{max} = 2$, $\omega_{max} = 1$, and $D_s = 0.5$ so that the vehicles follow a circular trajectory with radius $r = 1$ when applying v_{min} and ω_{max}. Assume now that the vehicles have the following

initial states

$$x_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T,$$
$$x_2 = \begin{bmatrix} (2r + D_s)\sin\psi & (2r + D_s)\cos\psi - 2r & \pi & 0 \end{bmatrix}^T,$$
$$x_3 = \begin{bmatrix} (2r + D_s)\sin\psi & 2r - (2r + D_s)\cos\psi & \pi & 0 \end{bmatrix}^T,$$

where $\psi = \arccos\left(\frac{D_s/2+2r}{2r+D_s}\right)$. These states are selected so that $h^1(x)$, $h^2(x)$, and $h^3(x)$ are all 0. See Figure 3.2 for the geometric setup that leads to these states being selected. In particular these states imply that vehicles 1 and 2 must turn right to avoid each other while vehicles 1 and 3 must turn left to avoid each other. At the same time, the states also imply that $h^1(x)$, $h^2(x)$, and $h^3(x)$ are all nonnegative so the system appears to be safe. We denote the safe sets associated with h^1, h^2, and h^3 as $\mathcal{C}^1$, $\mathcal{C}^2$, and $\mathcal{C}^3$, respectively.

Since $h^1(x) = h^2(x) = h^3(x) = 0$, the barrier constraints in (1.5) for $h^1(x)$ and $h^2(x)$ become

$$-0.4(v_1 + \omega_1 + v_2 + \omega_2) \geq 0 \tag{3.1}$$
$$0.4(-v_1 + \omega_1 - v_3 + \omega_3) \geq 0. \tag{3.2}$$

Although h^1 and h^2 are ZCBFs, these two constraints cannot be simultaneously satisfied for $v_i \in [v_{min}, v_{max}]$ and $|\omega_i| \leq \omega_{max}$. In particular, after substituting the minimum velocity $v_1 = v_2 = 1$, the first equation dictates that $\omega_1 + \omega_2 \leq -2$ (i.e., vehicles 1 and 2 must turn right). Similarly, the second equation dictates that vehicle 1 and 3 must turn left. The problem with this scenario is that vehicle 1 cannot simultaneously execute both nominal evading maneuvers (i.e., turn both left and right at the same time). To solve this problem, we will make sure that the evasive maneuver applied by a vehicle is the same for every barrier function. A geometric view of the

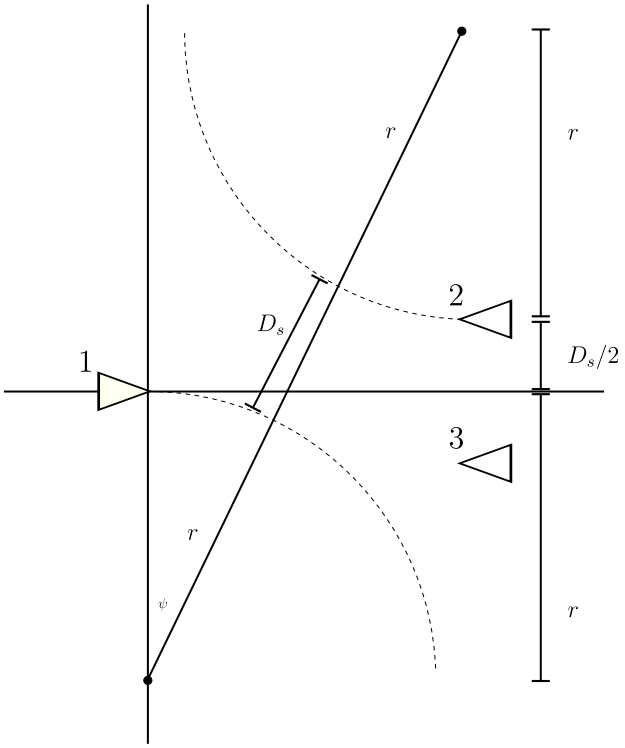


Figure 3.2: The geometric constraints that result in a case where safety cannot be maintained for three vehicles simultaneously. Vehicles 1 and 2 are placed so that only a hard right turn will keep them safe. Similarly, vehicles 1 and 3 are placed so that only a hard left turn will keep them safe. Finally, vehicles 2 and 3 are placed so they start D_s apart.

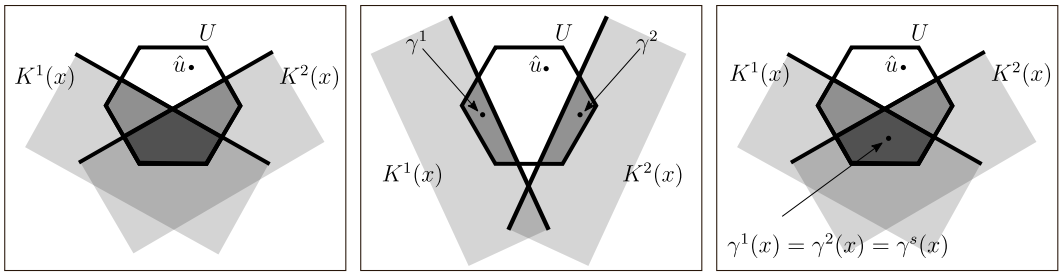


Figure 3.3: A geometric view of why having a set of individual barrier functions does not guarantee that a control input u exists to satisfy each associated constraint and how the shared evading maneuver assumption resolves this issue. In (a), multiple barrier function constraints are shown as half-spaces. To satisfy Corollary 1, a u must be selected that is in the intersection of $K^1(x)$, $K^2(x)$, and U. In (b), although there exists a u that is in the intersection of U and $K^1(x)$ as well as U and $K^2(x)$, as ensured by the fact that h^1 and h^2 are ZCBFs, there does not exist a u that is in the intersection of U, $K^1(x)$, and $K^2(x)$. This case corresponds to the specific scenario for the three vehicle collision avoidance problem in Figure 3.1c. In (c), the problem is resolved by the shared evading maneuver because $\gamma^s(x)$ satisfies each constraint.

general problem and its solution are shown in Figure 3.3.

Recall now the example of Section 2.1. In that example, the problem was that the candidate barrier function only calculated whether vehicles were currently in a collision but did not encode whether a future collision would occur. By discarding this candidate function in favor of $h_{straight}$ and h_{turn}, future collisions are taken into account. Put another way, the candidate barrier function of Section 2.1 rendered some states safe that were not actually safe because there was no overriding control value available to keep the system safe. This problem is alleviated by using a barrier function defined in (2.3) because γ is always available to keep the system safe. In the example of this section, we actually have a related problem. In particular, there are states where $h^1(x)$ and $h^2(x)$ are both nonnegative but the state is nevertheless unsafe. We therefore need to update h^1 and h^2 so that there not only exists a safe overriding control value to keep a single safety constraint value safe, but also it is the same safe overriding value that can be used across all safety constraints. As we will

see, the solution is to use the same γ across all barrier functions so that γ can be used to keep the system safe for all safety constraints.

3.2 Sufficient Conditions for Satisfying Multiple Safety Constraints

In order to solve the issues arising when vehicles have to simultaneously respect multiple constraints, we now extend the use of the constructive technique introduced in (2.3). In this section we extend the reasoning of [4] to the case of q constraints. Consider a nonlinear autonomous system

$$\dot{x} = f(x) \tag{3.3}$$

where f is locally Lipschitz. Then we have a similar definition to Definition 1 for autonomous systems.

Definition 2. [4] Given a set $\mathcal{C} \subset \mathbb{R}^n$ defined in (1.4) for a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$, the function h is called a zeroing barrier function (ZBF) defined on an open set $\mathcal{D}$ with $\mathcal{C} \subset \mathcal{D} \subset \mathbb{R}^n$, if there exists a Lipschitz continuous extended class $\mathcal{K}$ function α such that

$$L_f h(x) \geq -\alpha(h(x)),\ \forall x \in \mathcal{D}. \tag{3.4}$$

When there are q constraints, we consider the case of q barrier functions where each barrier function is denoted h^j on $\mathcal{D}^j$ with associated safe set $\mathcal{C}^j$ and admissible control space $K^j(x)$ for $x \in \mathcal{D}^j$ for $j \in \{1, \ldots, q\}$. We are interested in the conditions under which all safety constraints can be satisfied for all future times. In other words, under the assumption that $x(0) \in \mathcal{C}^j$ we want to show that $x(t) \in \mathcal{C}^j$ for all $t \geq 0$. Hence, we are interested in the forward invariance of the intersection of all the safe

sets, which motivates the following definitions

$$\mathcal{C}_\cap = \mathcal{C}^1 \cap \mathcal{C}^2 \cap \cdots \cap \mathcal{C}^q, \tag{3.5}$$

$$K_\cap(x) = \{u \in U \ : \ u \in K^1(x) \cap K^2(x) \cap \cdots \cap K^q(x)\}. \tag{3.6}$$

where $\mathcal{D}_\cap$ is an open superset of $\mathcal{C}_\cap$ and $x \in \mathcal{D}_\cap$. We can now present a multiple constraint analogue of Theorem 1 by following the same reasoning as [4].

Proposition 1. Given a dynamical system (3.3) and a set $\mathcal{C}_\cap$ defined by (3.5) for continuously differentiable functions $h^j : R^n \to R$ where h^j is a ZBF on D^j with $\mathcal{C}^j \subset D^j \subset R$ and $\frac{\partial h^j(x)}{\partial x} \neq 0$ for any $x \in \partial C_\cap$ where $h^j(x) = 0$, then $\mathcal{C}_\cap$ is forward invariant.

Proof. The proof is the same as that for Proposition 1 of [4], namely $\dot{h}^j(x) = -\alpha(x) \geq 0$ for any j such that $h^j(x) = 0$ so the result follows by Nagumo's Theorem [64]. We add the assumption that $\frac{\partial h^j(x)}{\partial x}$ is non-zero for all $x \in \partial\mathcal{C}_\cap$ such that $h^j(x) = 0$ to ensure that the tangent cone in Nagumo's Theorem is non-empty. □

Then for autonomous systems with dynamics (1.3), we have the following corollary of Theorem 1.

Corollary 1. Given a dynamical system (1.3) and a set $\mathcal{C}_\cap$ defined by (3.5) for continuously differentiable functions $h^j : R^n \to R$ where h^j is a ZCBF on D^j and $\frac{\partial h^j(x)}{\partial x} \neq 0$ for any $x \in \partial\mathcal{C}_\cap$ where $h^j(x) = 0$, then any Lipschitz continuous controller $u : \mathcal{D}_\cap \to U$ such that $u(x) \in K_\cap(x)$ will render the set $\mathcal{C}_\cap$ forward invariant.

Corollary 1 means that multiple safety constraints can be simultaneously satisfied provided the safety overriding controller satisfies the assumptions. Note an important difference between Theorem 1 and Corollary 1 however. In Theorem 1 it shows the constraints on a controller to keep the state within a single set $\mathcal{C}$. On the other hand, Corollary 1 gives constraints on a controller to keep the state within a possibly

much smaller set $\mathcal{C}_\cap$. In other words, the conclusion of Corollary 1 is a stronger statement and therefore hints that satisfying $u(x) \in K_\cap(x)$ in Corollary 1 is more difficult than satisfying $u(x) \in K(x)$ in Theorem 1. This suggests that to satisfy this stronger requirement we may need an additional assumption which we discuss in the next section. The assumption, which we call the shared evading maneuver assumption, ensures that $K_\cap(x)$ is non-empty by ensuring that the same nominal evading maneuver is in $K^j(x)$ for $k = 1, \ldots, q$. With this additional assumption, we can ensure that a safety overriding controller not only exists but also that it can be calculated in real-time.

3.3 The Shared Nominal Evading Maneuver Assumption

Suppose there are q constraints $\rho^j : \mathcal{D}^j \to \mathbb{R}$ $(j = 1, \ldots, q)$ that must be greater than or equal to 0 at all times. For the k agents with pairwise constraints, $q = k(k-1)/2$. We assume that for each constraint $j = 1, \ldots, q$, a locally Lipschitz nominal evading maneuver γ^j has been selected using the framework in (2.3). An example for fixed-wing UAVs with collision avoidance safety constraints is given in (2.9). Given q safety functions ρ^j and evading maneuvers γ^j for $j \in \{1, \ldots, q\}$, we construct q output functions h^j defined on $\mathcal{D}^j$ similarly to (2.3) where

$$h^j(x; \rho, \gamma) = \inf_{\tau \in [0,\infty)} \rho^j(\hat{x}^j(\tau)), \tag{3.7}$$

$$\hat{x}^j(\tau) = x + \int_0^\tau \dot{\hat{x}}^j(\eta) d\eta, \tag{3.8}$$

$$\dot{\hat{x}}^j(\tau) = f(\hat{x}^j(\tau)) + g(\hat{x}^j(\tau))\gamma^j(\hat{x}^j(\tau)). \tag{3.9}$$

Section 3.1 showed an example where $K_\cap(x)$ could be empty for some $x \in \mathcal{C}_\cap$. As a result, the assumptions of Corollary 1 could not be satisfied. In order to address the issue discussed in Section 3.1, we introduce an additional constraint on γ^j $(j = 1, \ldots, q)$ that all h^j are constructed from the same nominal evading maneuver.

Assumption 1. Given a dynamical system (1.3) and q output functions h^j defined in (3.7) for given safety functions ρ^j and evading maneuvers γ^j for $j \in \{1, \ldots, q\}$, the shared evading maneuver assumption holds if $\gamma^1(x) = \cdots = \gamma^q(x)$ for all $x \in \mathcal{D}_\cap$. The shared evading maneuver is denoted γ^s so that

$$\gamma^s(x) = \gamma^1(x) = \cdots = \gamma^q(x) \tag{3.10}$$

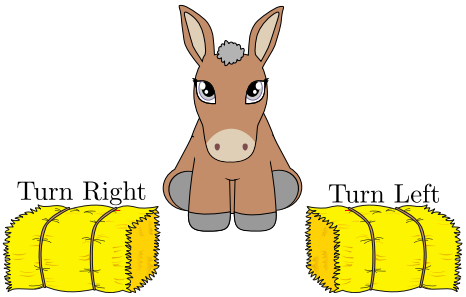
for all $x \in \mathcal{D}_\cap$.

Remark 3. This assumption requires that each h^j $(j = 1, \ldots, q)$ be constructed from the same nominal evading maneuver. Note, however, that this does not imply that each h^j must be constructed from the same safety function ρ^j.

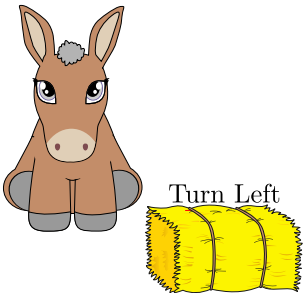
Remark 4. To gain an intuition of the importance of the shared evasive maneuver, consider an example unrelated to barrier functions, namely the paradox of Buridan's Donkey.[1] In this paradox, a donkey is placed perfectly in the middle between two completely identical stacks of hay. Due to the symmetry of the problem, there is no reason to choose one stack over the other, so the donkey will eventually starve. Similar to Buridan's Donkey, the situation in Figure 3.1 is an impossible choice because both safety factors are critical. There is no mechanism to prioritize one safety objective over the other because both objectives must be met at all times. However, an indirect solution that applies to both the paradox and Figure 3.1 is to avoid the problematic situation altogether. For Buridan's Donkey, that means only letting the donkey see one stack of hay (see Figure 3.4). For barrier functions, that means using only one evasive maneuver across all barrier functions. This means that the situation in Figure 3.1 will never occur because in this case $h^1(x)$ would not be safe since the shared evasive maneuver requires that both vehicles turn left.

The example in Section 3.1 does not satisfy Assumption 1 because $\gamma^1(x)$ and

[1] https://en.wikipedia.org/wiki/Buridan%27s_ass



(a)



(b)

Figure 3.4: (a) Buridan's Donkey cannot decide which stack of hay to eat so it starves. (b) By giving the donkey only one stack of hay, it is able to avoid starvation. The solution in (b) avoids starvation by not allowing the situation in (a) to occur in the first place. The same idea occurs with the shared evading maneuver. Rather than allowing situations where there is no control action available to satisfy two safety constraints, the shared evading maneuver simplifies the problem by ensuring a single solution exists across all safety problems.

$\gamma^2(x)$ defined in (3.1) are not the same. To enforce that the shared evasive maneuver assumption holds, one option is to change γ^1 so that

$$\gamma^1(x) = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}^T. \tag{3.11}$$

In other words, using γ^1 defined in (3.11) and γ^2 and γ^3 in (3.0b) implies an evasive maneuver where all vehicles turn left for each constraint. Another example where the shared nominal evading maneuver assumption holds is as follows:

$$\gamma^s(x) = \gamma^1(x) = \gamma^2(x) = \gamma^3(x) = \begin{bmatrix} 1 & 1 & 0 & 1.5 & 0 & 0 & 2 & -1 & 0 \end{bmatrix}^T.$$

In this case, $\gamma^s(x)$ encodes an evasive maneuver where vehicle 1 turns left with a linear velocity of 1, vehicle 2 stays straight with a linear velocity of 1.5, and vehicle 3 turns right with a linear velocity of 2. These three nominal evading maneuvers satisfy the shared evasive maneuver assumption because for all $x \in \mathcal{D}_\cap$, $\gamma^1(x) = \gamma^2(x) = \gamma^3(x)$.

To see the purpose of Assumption 1, we first examine the case of a single constraint. In particular, let h be defined in (2.3) and consider the role of γ in establishing that h is a ZCBF. From Definition 1, for h to be used for a barrier function, $K(x)$ must be nonempty for all $x \in \mathcal{D}$. With h defined as in (2.3), this property is satisfied by $\gamma(x)$ or a perturbation of $\gamma(x)$ for all $x \in \mathcal{D}$ (see Theorem 2). The analogue condition for multiple constraints is that $K_\cap(x)$ is non-empty for all $x \in \mathcal{D}_\cap$. If each h^j defined in (2.3) is a ZCBF and is constructed from γ^j then by similar reasoning to Theorem 2, $\gamma^j(x)$ or a perturbation of $\gamma(x)$ is in $K^j(x)$ for all $x \in \mathcal{D}_\cap$. This allows us to state a multiple constraint analogue to Theorem 2. In the following, we denote the inner product as $\langle L_g h^{j_1}(x), L_g h^{j_2}(x) \rangle$ for $j_1, j_2 \in \{1, \ldots, q\}$.

Theorem 3. Given a dynamical system (1.3) and a set $\mathcal{C}_\cap \subset \mathcal{D}_\cap$ defined in (3.5) for q continuously differentiable functions h^j defined in (3.7) with safety functions ρ^j and evading maneuvers γ^j where $k \in \{1, \ldots, q\}$, if h^j is a ZCBF for $k \in \{1, \ldots, q\}$ and Assumption 1 holds then $K_\cap(x)$ is non-empty for all $x \in \mathcal{C}_\cap$. If in addition, γ^s defined in (3.10) maps to the interior of U and for all $x \in \partial\mathcal{C}_\cap$, $\langle L_g h^{j_1}(x), L_g h^{j_2}(x) \rangle > 0$ for $j_1 \neq j_2$ and $j_1, j_2 \in \{1, \ldots, q\}$, then there is an open set that is a superset of $\mathcal{C}_\cap$ for which $K_\cap(x)$ is non-empty for all x in the open set.

Proof. To prove the first statement, note that it was shown in the proof of Theorem 2 that γ^s is in $K^j(x)$ for $j = 1, \ldots, q$ and $x \in \mathcal{C}_\cap$. To prove the second statement, note that we can use the same method as was used in the proof of Theorem 2 to find a vector $d(z)$ such that $h^j(z)$ satisfies (1.5) for all $z \in B(x, \mu)$ given $x \in \partial\mathcal{C}_\cap$. In particular, because $\langle L_g h^{j_1}(x), L_g h^{j_2}(x) \rangle > 0$, $L_g h^j(x) \neq 0$ for $j = 1, \ldots, q$ there exists a vector $d_{all}(x)$ such that $\langle d_{all}(x), L_g h^j(x) \rangle > 0$. We choose $d_{all}(x)$ with sufficiently small norm. Using the notation of the proof of Theorem 2, for sufficiently small μ, the projection of $d_{all}(x)$ onto $L_g h(z)$ will be in the direction of $L_g h(z)$ for $z \in B(x, \mu)$ because $L_g h(x)$ is continuous. $\square$

Remark 5. A geometric view of the problem introduced in Section 3.1 and its resolution via the shared evading maneuver assumption is shown in Figure 3.3.

Theorem 3 gives sufficient conditions for ensuring that multiple safety constraints can be satisfied for all times. Further, we can calculate the safe overriding control value in real time by adding constraints to (1.10). In particular, for q safety constraints, let $\hat{u} = \begin{bmatrix} \hat{u}_1^T & \hat{u}_2^T & \dots & \hat{u}_k^T \end{bmatrix}^T$ where $\hat{u}_i$ is the nominal input of vehicle i for $i = 1, \dots, k$. To emphasize that all h^j are constructed from γ^s, we write $h^{j,s}$ for each $j = 1, \dots, q$ as follows:

$$\begin{aligned} u^* = \min_{u \in \mathbb{R}^m} \ & \frac{1}{2} \|u - \hat{u}\|^2 \\ \text{s.t.} \quad & Au \geq b. \\ & L_f h^{j,s}(x) + L_g h^{j,s}(x) u + \alpha(h^{j,s}(x)) \geq 0 \qquad j \in \{1, \dots, q\}. \end{aligned} \tag{3.12}$$

In other words, safety for multiple constraints can be guaranteed with a fast online calculation by satisfying the shared evading maneuver assumption and adding constraints to the QP.

3.4 Conclusion

In this chapter we have shown that merely being able to ensure individual safety constraints are satisfied does not mean all constraints can be satisfied simultaneously. To verify this insight we presented a particular example where it is impossible to satisfy multiple constraints. This occurs where a vehicle must turn left to avoid one of its neighbors but must turn right to avoid its other neighbor. The example hints that always using a consistent evasive maneuver for all safety objectives can resolve this problem. We encode this intuition in the shared evading maneuver assumption and show that under this assumption we can ensure that multiple safety constraints can be satisfied for all future time. We therefore add this conclusion to the table of

Table 3.1: A list of assumptions required to ensure system safety after adding the shared evading maneuver

Assumption	Effect of Removing Assumption
One safety constraint	still safe but additional assumption required (shared evading maneuver)
Unlimited Communication	see Chapter 4
Infinite Range Sensors	see Chapter 5
Known dynamics model	see Chapter 6

assumptions in Table 3.1.

CHAPTER 4

RELAXING COMMUNICATION CONSTRAINTS

In Chapter 2 we showed that by specifying an evasive maneuver, we could guarantee the system will stay safe for all times. Chapter 3 then expanded on this concept by showing that if the same evasive maneuver is used to across multiple safety constraints then all the constraints could be satisfied for all times. In other words, the previous chapter relaxed one assumption from Chapter 2, namely that there was only one safety objective. In this chapter we remove another assumption, namely that the vehicles can communicate their control signals. In particular, we show that the calculation of the admissible control space requires knowledge of all vehicle control values and therefore implies a significant communication overhead in the case of a swarm of aircraft. Thus, in this chapter we remove the assumption that vehicles can communicate their low level control signals and show that we can still ensure safety in this case.

Consider now how to implement the conclusions of Chapters 2 and 3 for UAV collision avoidance. In particular, after specifying a dynamics model, we construct h^j in (2.3) by using the same evasive maneuver γ for each h^j. After that we calculate the QP in (3.12) at every timestep to ensure the control value applied to the vehicles is not only safe but also as close as possible to the original control value $\hat{u}$.

To make such a calculation, the vehicles then need access to the following: the evasive maneuver of the other vehicle γ^s and safety objectives ρ^j, the state x, and the nominal control value $\hat{u}$. We assume that the evasive maneuver γ^s and safety functions ρ^j are known at deployment time via a locker room agreement. We consider the case where x is not always known in the next chapter. In this chapter we consider $\hat{u}$.

Recall that $\hat{u} = \begin{bmatrix} \hat{u}_1 & \hat{u}_2 \end{bmatrix}^T$ is the nominal control value that is the concatenation of both vehicle's individual nominal control values. In other words, for the first vehicle to

calculate the QP in (3.12), it must know the nominal control value of the other vehicle. This implies a communication link between the two vehicles that requires frequently communicating this value without any delays. However, frequently communicating this signal when there are many vehicles may reduce throughput for other important messages or introduce communication delays because a network can only support a limited number of bits per second through a network. Thus, we show how to ensure safety constraints can be satisfied by reformulating the QP so that the vehicles can calculate a safe control signal without requiring each other's nominal control input. However, we continue to assume that each vehicle can sense the state of every other vehicle. See Chapter 5 for relaxing this assumption.

The approach discussed below is to start with the ZCBF constraint (1.5) and note that the components $L_g h(x)$ and u are vectors. Hence their inner product is the sum of individual terms. The insight below is that we can then break up the individual terms of $L_g h(x)u$ into terms where some terms only depend on the first aircraft's control input while the other terms only depend on the second aircraft's control input. This results in breaking (1.5) into two constraints where the first constraint only depends on the first individual aircraft's control value and similarly for the second constraint. We finally exploit the fact that γ can be similarly decomposed and can then be used to ensure that both individual constraints can always be satisfied. The result is each individual aircraft can calculate a QP that only depends on its own control value that nevertheless ensures safety.

4.1 Limited Communication for Two Vehicles

We start by considering the two vehicle case and then generalize to the k vehicle case. Let $\gamma^s = \begin{bmatrix} \gamma_1^{s^T} & \gamma_2^{s^T} \end{bmatrix}^T$ be the shared evading maneuver where γ_1^s is the part of γ^s that is applied to vehicle 1 and therefore has the same size as u_1. Define γ_2^s similarly for vehicle 2. Similarly decompose b in (1.10c) as $b = \begin{bmatrix} b_1^T & b_2^T \end{bmatrix}^T$ and $L_g h^{j,s}(x)$ as

$L_g h^{j,s}(x) = \begin{bmatrix} [L_g h^{j,s}(x)]_1^T & [L_g h^{j,s}(x)]_2^T \end{bmatrix}^T$. Further, let A in (1.10c) be block diagonal with block entries A_1 and A_2 so that $A_i u_u \geq b_i$ represents the actuator constraint for vehicle i for $i = 1, 2$.

We want to find a way of calculating u_1 and u_2 such that $u = \begin{bmatrix} u_1^T & u_2^T \end{bmatrix}^T$ satisfies $Au \geq b$ and $u \in K^j(x)$ for all $x \in \mathcal{D}$ where the calculation for u_1 does not require knowledge of $\hat{u}_2$ or the final value for u_2. Similarly, we want to calculate u_2 without knowledge of $\hat{u}_1$ or u_1. This is a trivial requirement for actuator constraints since $A_i u_i \geq b_i$ for $i = 1, 2$ if and only if $Au \geq b$. However, the constraint that $u \in K_\cap(x)$ involves both u_1 and u_2 so we reformulate it. We leverage the fact that

$$L_g h(x) u = \begin{bmatrix} [L_g h(x)]_1 & [L_g h(x)]_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = [L_g h(x)]_1 u_1 + [L_g h(x)]_2 u_2$$

so that

$$\begin{aligned} 0 &\leq L_f h^{j,s}(x) + L_g h^{j,s}(x) u + \alpha(h^{j,s}(x)) && (4.1) \\ &= \kappa_1(x, u_1) + \kappa_2(x, u_2) && (4.2) \end{aligned}$$

where

$$\begin{aligned} \kappa_1(x, u_1) = {} & L_f h^{j,s}(x) + [L_g h^{j,s}(x)]_1 u_1 + \alpha(h^{j,s}(x)) + [L_g h^{j,s}(x)]_2 \gamma_2^s \\ & - \frac{1}{2}(L_f h^{j,s}(x) + L_g h^{j,s}(x) \gamma^s + \alpha(h^{j,s}(x))) \end{aligned} \quad (4.3)$$

and

$$\begin{aligned} \kappa_2(x, u_2) = {} & L_f h^{j,s}(x) + [L_g h^{j,s}(x)]_2 u_2 + \alpha(h^{j,s}(x)) + [L_g h^{j,s}(x)]_1 \gamma_1^s \\ & - \frac{1}{2}(L_f h^{j,s}(x) + L_g h^{j,s}(x) \gamma^s + \alpha(h^{j,s}(x))). \end{aligned} \quad (4.4)$$

Notice that κ_1 is not a function of u_2 and κ_2 is not a function of u_1. In other words, if we can select u_1 and u_2 such that $\kappa_1(x, u_1) \geq 0$ and $\kappa_2(x, u_2) \geq 0$ then $u = \begin{bmatrix} u_1^T & u_2^T \end{bmatrix}^T \in K_\cap(x)\ \forall x \in \mathcal{D}$. We must then show that $K_\cap(x)$ is non-empty for all in an open set that is larger than $\mathcal{C}_\cap$. For $x \in \mathcal{C}_\cap$, this can be done by letting $u_1 = \gamma_1^s(x)$ and $u_2 = \gamma_2^s(x)$ and noting that this implies

$$\kappa_1(x, \gamma_1^s) + \kappa_2(x, \gamma_2^s) = L_f h^{j,s}(x) + L_g h^{j,s}(x)\gamma^s + \alpha(h^{j,s}(x)) \geq 0.$$

For $x \notin \mathcal{C}_\cap$, a perturbation of $\gamma_1^s(x)$ and $\gamma_2^s(x)$ using a similar method as shown in the proof of Theorem 3 suffices. In other words, we can find u without vehicle 1 needing to know $\hat{u}_2$ or u_2 and similarly for vehicle 2. Each vehicle i ($i = 1, 2$) could then calculate the following QP:

$$\begin{aligned} u^* = \min_{u \in \mathbb{R}^{m_i}} \ & \frac{1}{2} \left\| u - \hat{u}_i \right\|^2 \\ \text{s.t.} \quad & A_i u_i \geq b_i \\ & \kappa_i(x, u_i) \geq 0. \end{aligned} \tag{4.5}$$

Note that $\kappa_i(x, u_i)$ is linear in u_i. To summarize, we started with the ZCBF constraint in (1.5), and split the $L_g h(x) u$ term into two components that depend on u_1 and u_2 respectively. Because each vehicle can always use the component of γ that corresponds to its own control input, there is always a control input to keep the vehicles safe. Further, this calculation can again be done with a QP so there is a fast online calculation of the safe control input available at all times.

4.2 Limited Communication for k Vehicles

We now generalize the discussion in Section 4.1 to k vehicles. Let $\gamma^s = \begin{bmatrix} \gamma_1^{s^T} & \dots & \gamma_k^{s^T} \end{bmatrix}^T$, where γ_i^s maps to vectors of the same size as u_i for $i = 1, \dots, k$ with similar decom-

position for $b = \begin{bmatrix} b_1^T & \cdots & b_k^T \end{bmatrix}^T$ and $L_g h^{j,s}(x) = \begin{bmatrix} [L_g h^{j,s}(x)]_1^T & \cdots & [L_g h^{j,s}(x)]_k^T \end{bmatrix}^T$. Further, assume A in (1.10c) is block diagonal with block entries A_i for $i = 1, \ldots, k$ where A_i is a $m_i \times m_i$ matrix. This assumption means that actuator constraints are not coupled between vehicles.

The main difference when there $k > 2$ than when $k = 2$ is that not every vehicle affects every constraint. For instance, in the example of Section 3.1, vehicle 3 does not affect h^1 since h^1 is only designed to keep vehicles 1 and 2 from colliding. For this reason we denote

$$\mathcal{V}^j = \{i \in \{1, \ldots, k\} : \exists x \in \mathcal{D} \text{ s.t. } [L_g h^{j,s}(x)]_i \neq 0_{m_i}\}$$

where 0_{m_i} is the zero vector in $\mathbb{R}^{m_i}$ and $j \in \{1, \ldots, q\}$. $\mathcal{V}^j$ represents the set of vehicles whose control input affects the time derivative of h^j for some $x \in \mathcal{D}$. We let $|\mathcal{V}^j|$ denote the cardinality of $\mathcal{V}^j$, and note that for the case of pairwise collision avoidance, $|\mathcal{V}^j| = 2$ for all $j = 1, \ldots, q$. In the example with three vehicles in Section 3.1, $\mathcal{V}_1 = \{1, 2\}$, $\mathcal{V}_2 = \{1, 3\}$, $\mathcal{V}_3 = \{2, 3\}$. We also denote $\mathcal{S}_i = \{j \in \{1, \ldots, q\} \ : \ i \in \mathcal{V}^j\}$ so that $\mathcal{S}_i$ is the set of safety constraint indices where u_i has an effect on the time derivative of the associated barrier function for some $x \in \mathcal{D}$. For the three vehicle example of Section 3, $\mathcal{S}_1 = \{1, 2\}$, $\mathcal{S}_2 = \{1, 3\}$, $\mathcal{S}_3 = \{2, 3\}$.

Finally, we denote $u_{\setminus i} = \begin{bmatrix} u_1^T & \cdots & u_{i-1}^T & u_{i+1}^T & \cdots & u_k^T \end{bmatrix}^T$, with similar definitions for $\gamma_{\setminus i}^s$, $\hat{u}_{\setminus i}$, and $[L_g h^{j,s}(x)]_{\setminus i}$. With the above definitions, we can now state a limited communication analogue for the admissible control space in (1.7). In analogy with the admissible control space in (1.7), the limited communication admissible control space for constraint j ($j = 1, \ldots, q$) and vehicle i ($i \in \mathcal{V}^j$) is defined as

$$\mathcal{K}_i^j(x) = \Big\{ u_i \in U_i \ : \ 0 \leq L_f h^{j,s}(x) + [L_g h^{j,s}(x)]_i u_i + \alpha(h^{j,s}(x)) + [L_g h^{j,s}(x)]_{\setminus i} \gamma_{\setminus i}^s(x) - \frac{|\mathcal{V}^j| - 1}{|\mathcal{V}^j|} \Big(L_f h^{j,s}(x) + L_g h^{j,s}(x) \gamma^s(x) + \alpha(h^{j,s}(x)) \Big) \Big\}. \quad (4.6)$$

Notice that the constraint of $\mathcal{K}_i^j(x)$ is the same as the constraint $\kappa_1(x, u_1) \geq 0$ in (4.3) when $i = 1$ and $k = 2$ so that it is a generalization of the situation when there are two vehicles. However, $\mathcal{K}_i^j$ only encodes controls that satisfy one safety constraint so we now take the intersection over all j where aircraft i affects the j^{th} constraint. The limited communication admissible control space for vehicle i is then $\mathcal{K}_i(x) = \bigcap_{l \in \mathcal{S}_i} \mathcal{K}_i^l(x)$. Although $\mathcal{K}_i(x)$ is the set admissible controls that encode safety for all constraints, it is only specific to one aircraft, namely aircraft i. We therefore concatenate the vectors to arrive at a set of admissible controls similar that results in a set that is $K_\cap$. The overall limited communication admissible control space is then

$$\mathcal{K}(x) = \left\{ u = \begin{bmatrix} u_1^T & \dots & u_k^T \end{bmatrix}^T \in U \ : \ u_i \in \mathcal{K}_i(x) \ \forall i \in \{1, \dots, k\} \right\}.$$

We now show that $\mathcal{K}(x)$ can be used in a similar way as $K_\cap(x)$ was used in Theorem 3 for ensuring safety.

Theorem 4. Given a dynamical system (1.3) and a set $\mathcal{C}_\cap \subset \mathcal{D}_\cap$ defined in (3.5) for q continuously differentiable functions h^j defined in (3.7) with safety functions ρ^j and evading maneuvers γ^j where $k \in \{1, \dots, q\}$, if h^j is a ZCBF for $k \in \{1, \dots, q\}$ and Assumption 1 holds then $\forall x \in \mathcal{D}_\cap$, $\mathcal{K}(x) \subseteq K_\cap(x)$. Further, $\mathcal{K}(x)$ is non-empty for all $x \in \mathcal{C}_\cap$. If in addition, γ^s maps to the interior of U and for all $x \in \partial\mathcal{C}_\cap$, $\langle [L_g h^{j_1}(x)]_i, [L_g h^{j_2}(x)]_i \rangle > 0$ for $j = 1, \dots, q$ and $i = 1, \dots, k$ and $j_1 \neq j_2$ and $j_1, j_2 \in \{1, \dots, q\}$, then there is an open set that is a superset of $C_\cap$ for which $\mathcal{K}(x)$ is non-empty for all x in the open set.

Proof. For the first statement, assume $u \in \mathcal{K}(x)$ so that $u_i \in \mathcal{K}_i(x) \ \forall i \in \{1, \dots, k\}$. This means that $A_i u_i \geq b_i$ so that, because A is block diagonal, $Au \geq b$. Further, it

means that for any constraint $j = 1, \ldots, q$ and any $i \in \mathcal{V}^j$,

$$\begin{aligned} 0 \leq L_f h^{j,s}(x) + [L_g h^{j,s}(x)]_i u_i + \alpha(h^{j,s}(x)) + [L_g h^{j,s}(x)]_{\setminus i} \gamma^s_{\setminus i}(x) \\ -\frac{|\mathcal{V}^j| - 1}{|\mathcal{V}^j|}\Big(L_f h^{j,s}(x) + L_g h^{j,s}(x)\gamma^s(x) + \alpha(h^{j,s}(x))\Big). \end{aligned} \tag{4.7}$$

To simplify (4.7), note that by definition, $[L_g h^{j,s}(x)]_i = 0_{m_i}$ for $i \neq \mathcal{V}^j$ so that

$$\begin{aligned} \sum_{i \in \mathcal{V}^j} [L_g h^{j,s}(x)]_i u_i &= \sum_{i \in \{1,\ldots,k\}} [L_g h^{j,s}(x)]_i u_i \\ &= L_g h^{j,s}(x) u. \end{aligned} \tag{4.8}$$

Using (4.8) in the following then yields

$$\begin{aligned} \sum_{i \in \mathcal{V}^j} [L_g h^{j,s}(x)]_{\setminus i} \gamma^s_{\setminus i}(x) &= \sum_{i \in \mathcal{V}^j} \big(L_g h^{j,s}(x)\gamma^s(x) - [L_g h^{j,s}(x)]_i \gamma^s_i(x)\big) \\ &= |\mathcal{V}^j| L_g h^{j,s}(x)\gamma^s(x) - \sum_{i \in \mathcal{V}^j} [L_g h^{j,s}(x)]_i \gamma^s_i(x) \\ &= |\mathcal{V}^j| L_g h^{j,s}(x)\gamma^s(x) - L_g h^{j,s}(x)\gamma^s(x) \\ &= (|\mathcal{V}^j| - 1) L_g h^{j,s}(x)\gamma^s(x). \end{aligned} \tag{4.9}$$

Summing (4.7) over $i \in \mathcal{V}^j$ and using (4.8) and (4.9) yields

$$\begin{aligned} 0 &\leq |\mathcal{V}^j| L_f h^{j,s}(x) + L_g h^{j,s}(x) u + |\mathcal{V}^j| \alpha(h^{j,s}(x)) + (|\mathcal{V}^j| - 1) L_g h^{j,s}(x)\gamma^s(x) \\ &\quad -(|\mathcal{V}^j| - 1)\Big(L_f h^{j,s}(x) + L_g h^{j,s}(x)\gamma^s(x) + \alpha(h^{j,s}(x))\Big) \\ &= L_f h^{j,s}(x) + L_g h^{j,s}(x) u + \alpha(h^{j,s}(x)). \end{aligned}$$

Since this is true for all $j = 1, \ldots, q$, $u \in K_\cap(x)$. Then $\mathcal{K}(x) \subseteq K_\cap(x)$ for all $x \in \mathcal{C}_\cap$.

Consider now the second statement, namely that $\gamma^s \in \mathcal{K}(x)$. For $j = 1, \ldots, q$,

consider any $i \in \mathcal{V}^j$ and let $u_i = \gamma_i^s$. Then

$$\begin{aligned}
&L_f h^{j,s}(x) + [L_g h^{j,s}(x)]_i u_i + \alpha(h^{j,s}(x)) + [L_g h^{j,s}(x)]_{\setminus i} \gamma^s_{\setminus i}(x) \\
&- \frac{|\mathcal{V}^j| - 1}{|\mathcal{V}^j|} \Big(L_f h^{j,s}(x) + L_g h^{j,s}(x) \gamma^s(x) + \alpha(h^{j,s}(x)) \Big) \\
&\quad = \frac{1}{|\mathcal{V}^j|} \left(L_f h^j(x) + L_g h^j(x) \gamma^s(x) + \alpha(h^j(x)) \right) \\
&\quad \geq 0.
\end{aligned}$$

The inequality is true because $x \in \mathcal{C}_\cap$ implies $\alpha(h^{j,s}(x)) \geq 0$. See the proof for Theorem 2 for why $L_f h^{j,s}(x) + L_g h^{j,s}(x) \gamma^s(x) \geq 0$. Then $\gamma_i^s \in \mathcal{K}_i^j$ for any $j = 1, \ldots, q$ and $i \in \mathcal{V}^j$. Then $\gamma_i^s \in \mathcal{K}_i$. Then $\gamma^s(x) \in \mathcal{K}(x)$.

Finally, the last statement where $\mathcal{K}(x)$ is nonempty for all x in an open set that is a superset of $\mathcal{C}_\cap$ follows similarly to the proof of Theorem 3. □

Theorem 4 means that each aircraft can choose its control value to be in $\mathcal{K}_i(x)$ because the resulting control value across all aircraft will be in $\mathcal{K}(x)$ which is a subset of $K_\cap(x)$. Further, because of the shared evading maneuver assumption, $\mathcal{K}_i(x)$ will always be non-empty. This allows us to then write a QP similar to (3.12) but without requiring knowledge of other agents' low level control values as follows:

$$\begin{aligned}
u_i^* = \min_{u_i \in \mathbb{R}^{m_i}} \ & \frac{1}{2} \|u_i - \hat{u}_i\|^2 \qquad (4.10) \\
\text{s.t.} \quad & A_i u_i \geq b_i \\
& L_f h^{j,s}(x) + [L_g h^{j,s}(x)]_i u_i + \alpha(h^{j,s}(x)) + [L_g h^{j,s}(x)]_{\setminus i} \gamma^s_{\setminus i}(x) \\
& \quad - \frac{|\mathcal{V}^j| - 1}{|\mathcal{V}^j|} \Big(L_f h^{j,s}(x) + L_g h^{j,s}(x) \gamma^s(x) + \alpha(h^{j,s}(x)) \Big) \geq 0 \qquad j \in \mathcal{S}_i.
\end{aligned}$$

.

4.3 A Comparison of Safety With and Without Communication

We note that the solution from the centralized QP (3.12) may be different than the solution from the limited communication QPs (4.10) because $\mathcal{K}(x)$ may be a strict subset of $K_\cap(x)$. To see this, let $k = 2$, $q = 1$, $L_f h(x) = 0$, $\alpha(h(x)) = 0$, $m_1 = m_2 = 1$, $[L_g h(x)]_2 \gamma_2^s(x) = -1$, and $[L_g h(x)]_1 \gamma_1^s(x) = 1$. Then the barrier function constraint in (4.10) becomes $[L_g h(x)]_1 u_1 \geq 1$, while the barrier function constraint in (3.12) becomes $L_g h(x) u \geq 0$. Since $u_1 = 0$ is feasible for the latter but not the former equation, we do not have that $\mathcal{K}(x) = K_\cap(x)$. Because $\mathcal{K}(x) \subset K_\cap(x)$, it may be that the total cost of each vehicle calculating (4.10) is higher than the centralized calculation (3.12). In other words, the calculated safe control may not be as close to the nominal control signal in a least squares sense when using (4.10) as opposed to (3.12). Nevertheless, in either case of (3.12) or (4.10), a solution exists to the corresponding QP such that $u \in K_\cap$.

Although a specific example has been given to show that $\mathcal{K}(x)$ can be a strict subset of $K_\cap(x)$, consider more generally why this is the case by analyzing (4.2) again. The generalization of this equation is the sum over $\mathcal{V}^j$ of equation (4.7) but for simplicity we consider (4.2) and note the same conclusion holds in the more general case. Note then that in the limited communication case where vehicle 1 does not know u_2 and vehicle 2 does not know u_1 that in order to ensure the inequality holds, we must require that $\kappa_1(x, u_1) \geq 0$ and $\kappa_2(x, u_2) \geq 0$. However, when there is unlimited communication, we only need to have that $\kappa_1(x, u_1) + \kappa_2(x, u_2) \geq 0$. In other words, when there is unlimited communication, we can have that $\kappa_1(x, u_1) < 0$ or $\kappa_2(x, u_2) < 0$ provided that their sum is positive. This means that limited communication is introducing an additional constraint on the set of available control inputs and it is therefore not surprising that the set of controls that satisfies the safety constraint using limited communication, namely $\mathcal{K}(x)$, is smaller than the set

of controls that satisfy the safety constraint when there is unlimited communication, namely $K_\cap(x)$.

Another difference between the limited communication (4.10) and the centralized (3.12) QPs is how the size of the optimization variable and number of constraints vary with the number of vehicles k. In the centralized approach (3.12) the size of the optimization variable grows linearly with k while the number of constraints grows quadratically. On the other hand, in the limited communication QP (4.10), the size of the optimization variable and number of constraints are constant and linear, respectively.

4.4 Simulation of 20 Vehicles With Limited Communication

We now repeat the scenario discussed in Section 2.5 but consider $k = 20$ vehicles. For the scenario where h is constructed from γ_{turn}, we use $\begin{bmatrix} v & \omega & 0 & v & \omega & 0 \end{bmatrix}^T$ where $v = 0.9v_{min} + 0.1v_{max}$ and $\omega = 0.9\omega_{max}$. For the scenario where h is constructed from $\gamma_{straight}$, we let $\gamma^i = \begin{bmatrix} (1+0.01i)v & 0 & 0 \end{bmatrix}^T$ so that each vehicle uses a different translational velocity as is required to ensure differentiability of h (see Section 2.3.2). Note that this does not violate the shared evading maneuver assumption because $\gamma^s = \begin{bmatrix} (\gamma^1)^T & \cdots & (\gamma^k)^T \end{bmatrix}^T$. Additionally, we let $\psi = 0$ and $\psi = 25^\circ$ in the scenario where h is constructed from γ_{turn} and $\gamma_{straight}$, respectively. Offsetting the initial orientation 25° from pointing at the origin is required so that the vehicles can start in the safe set when using $\gamma_{straight}$. Screenshots for the case of γ_{turn} and $\gamma_{straight}$ are shown in Figures 4.1 and 4.2, respectively. A video of the resulting behavior is available in [65]. Quantitative results for both scenarios are shown in Figures 4.3 and 4.4 which shows similar outputs to the results for the two vehicle simulation shown in Figures 2.4 and 2.5. We also compare the approach of this paper to a navigation function from [7] in Figures 4.3 and 4.4. Note that when using a navigation function the vehicles begin the evasive maneuver earlier than when the collision avoidance algorithm is based on

Table 4.1: A list of assumptions required to ensure system safety after adding the shared evading maneuver and removing the need to communicate control values.

Assumption	Effect of Removing Assumption
One Safety Constraint	still safe but additional assumption required (shared evading maneuver)
Unlimited Communication	still safe but overriding control value may deviate more significantly from nominal control value
Infinite Range Sensors	see Chapter 5
Known Dynamics Model	see Chapter 6

a barrier function constructed from γ_{turn}. Using a less aggressive α function, such as a linear function with a small coefficient instead of a cubic function, may have caused the behavior from the barrier function override to similarly override earlier. A comparison of potential and barrier functions can also be found in [66]. Also note that the pairwise distance between all vehicles are kept above the minimum safety distance D_s while satisfying actuator constraints.

4.5 Conclusion

In this chapter we investigated a practical implementation consideration involving limited communication. We found that implementing Theorem 3 in the QP (3.12) required both infinite range sensing and high frequency communication of control values between aircraft. By splitting the safety constraint into groups of values that only depend on individual aircraft, we were able to reformulate the QP (3.12) to only depend on individual aircraft control values. This means we can ensure system safety without requiring the overhead of frequent communication of low level control values. However, we also find that while safety can be ensured, it may be that the resulting overriding control value may deviate more significantly from the nominal control value than when allowing for more communication. The summary of this discussion is therefore added to Table 4.1. We address limited sensing in the next chapter.

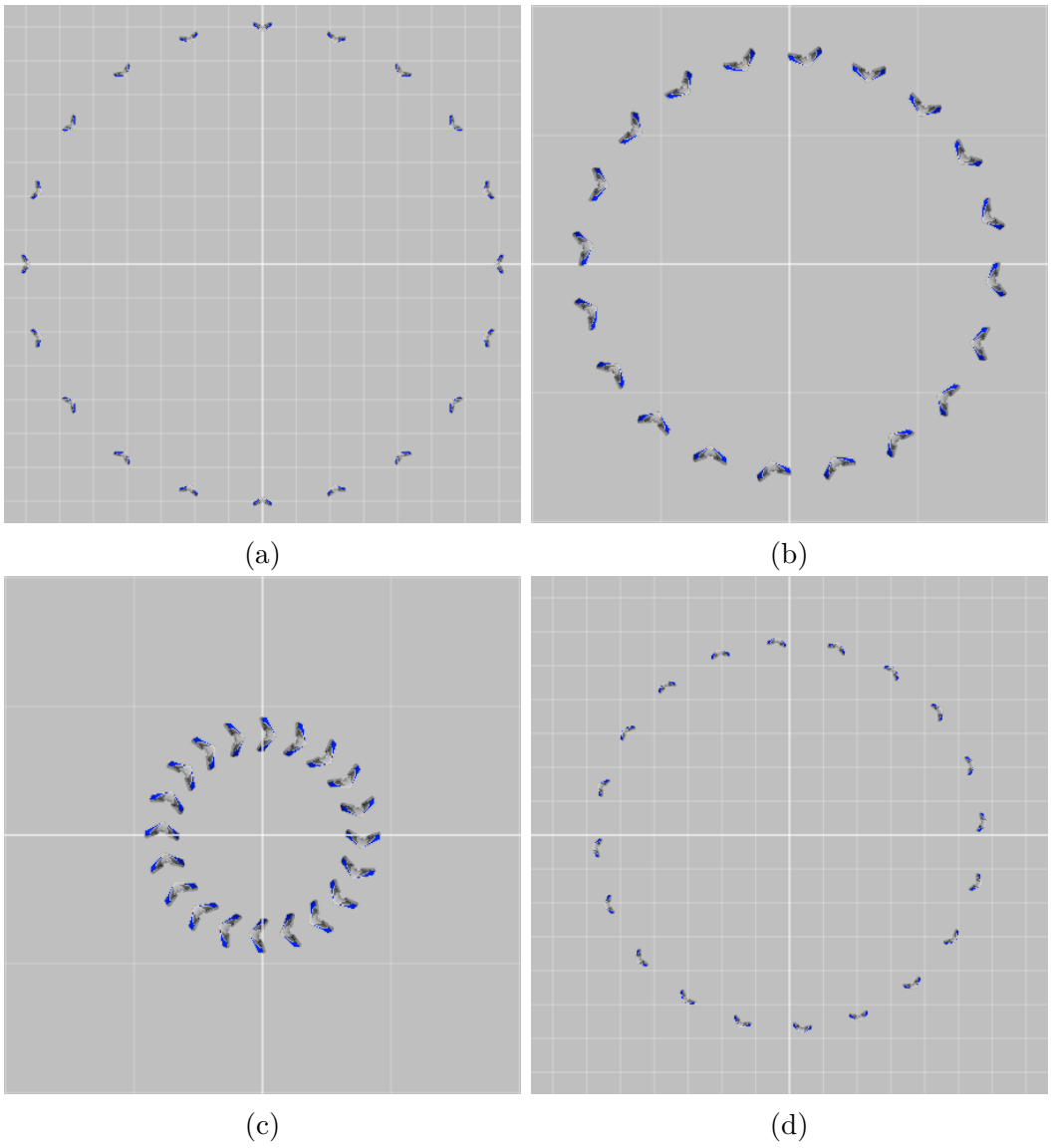


Figure 4.1: A demonstration of 20 fixed-wing vehicles applying barrier functions to ensure collisions are avoided when constructing h defined in (2.3) by γ_{turn}. (a) The starting position of 20 vehicles. (b) The vehicles approach the origin and begin avoidance behavior around 50 meters away from the origin. (c) The vehicles circle the origin. (d) The vehicles reach approach their target position.

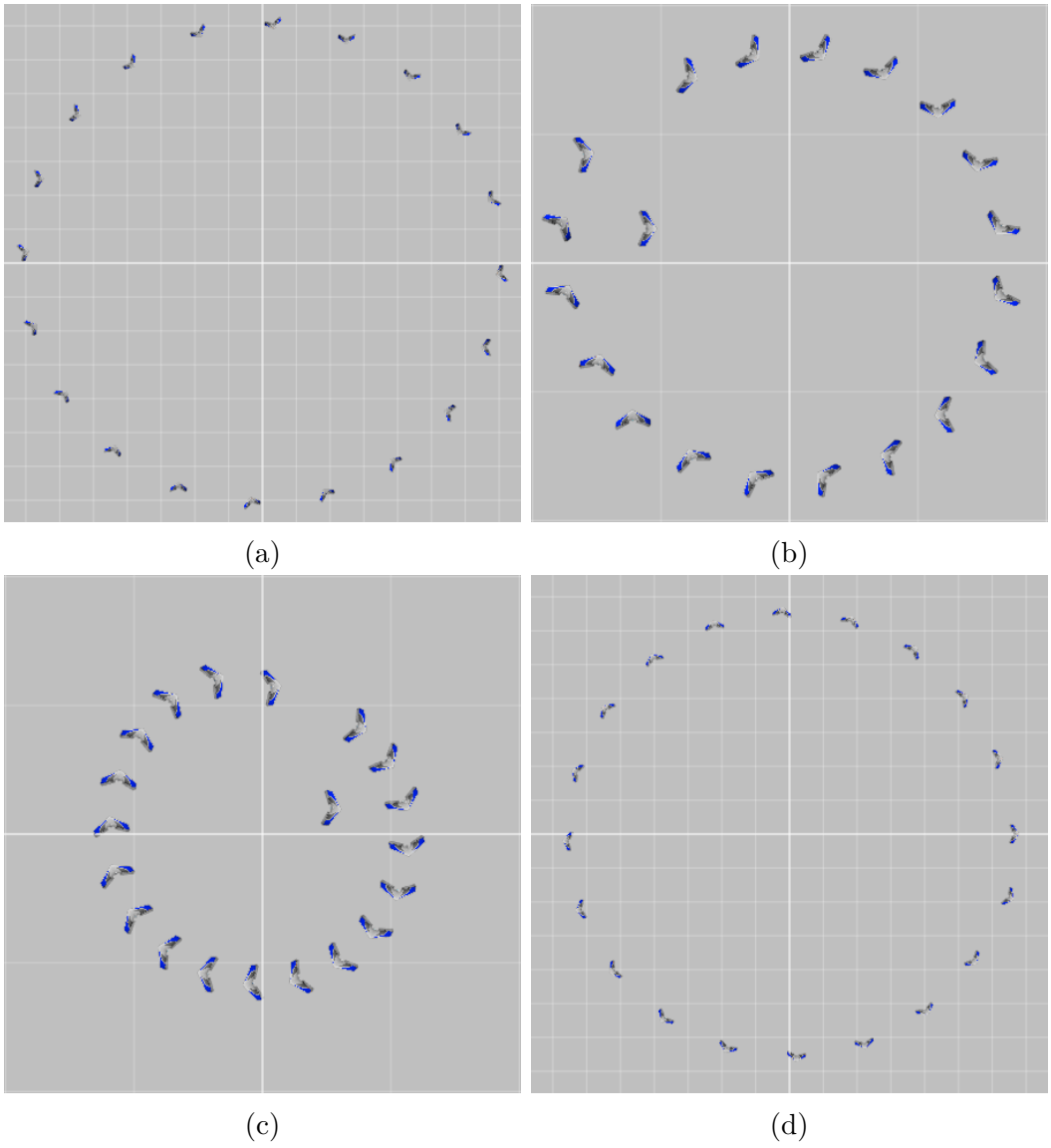
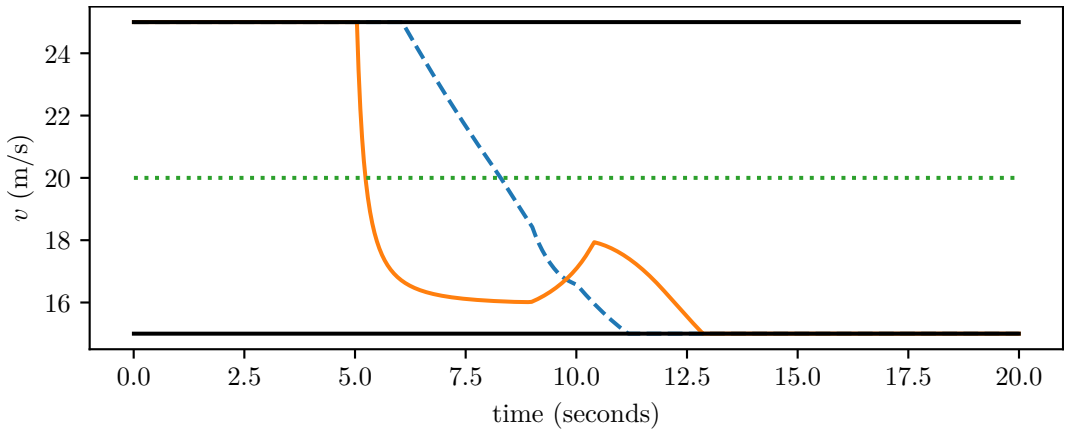
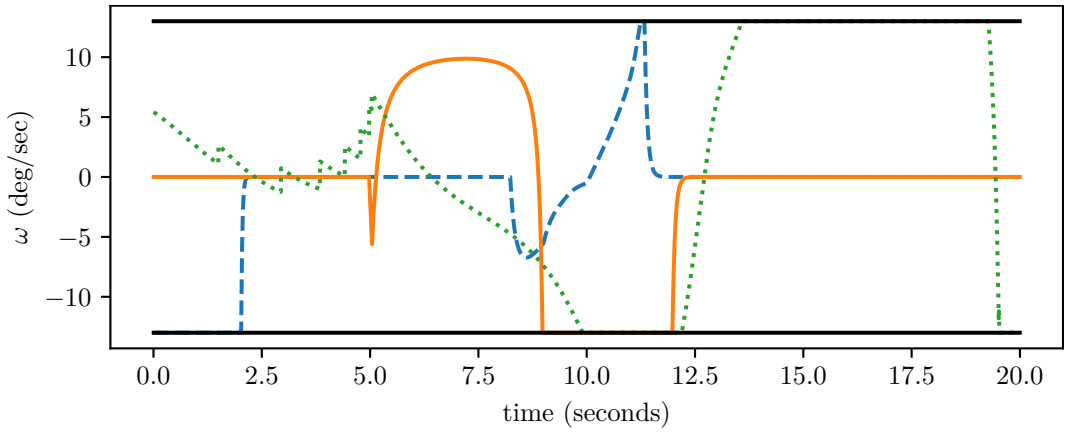


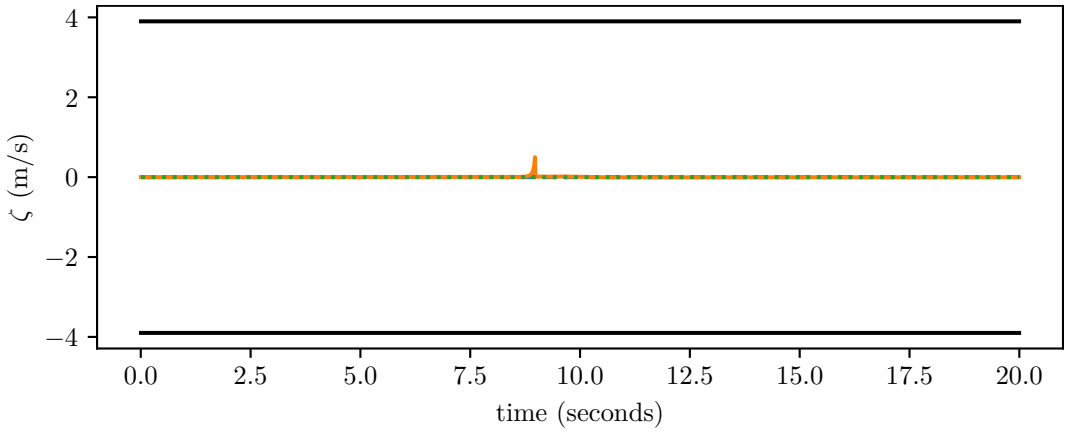
Figure 4.2: A demonstration of 20 fixed-wing vehicles applying barrier functions to ensure collisions are avoided when constructing h defined in (2.3) by $\gamma_{straight}$. (a) The starting position of 20 vehicles. (b) The vehicles approach the origin and begin avoidance behavior around 50 meters away from the origin. (c) The vehicles circle the origin. (d) The vehicles reach approach their target position. The asymmetry is due to the fact that the vehicles have different speeds for their nominal evading maneuvers. As the speed for the nominal maneuvers approaches the same value the result is a more symmetric pattern.



(a)



(b)



(c)

Figure 4.3: Control outputs for the scenario with 20 fixed-wing vehicles. The blue dashed and orange solid lines are the output of the scenario where h is constructed from $\gamma_{straight}$ and γ_{turn}, respectively. The green dotted line is the output from using a navigation function. Vehicle 1 velocity and turn rates are shown to be within the actuator limits in (a), (b), and (c) for velocity, turn rate, and altitude rate, respectively. The navigation function starts avoiding a collision well in advance of when using h constructed from γ_{turn}.

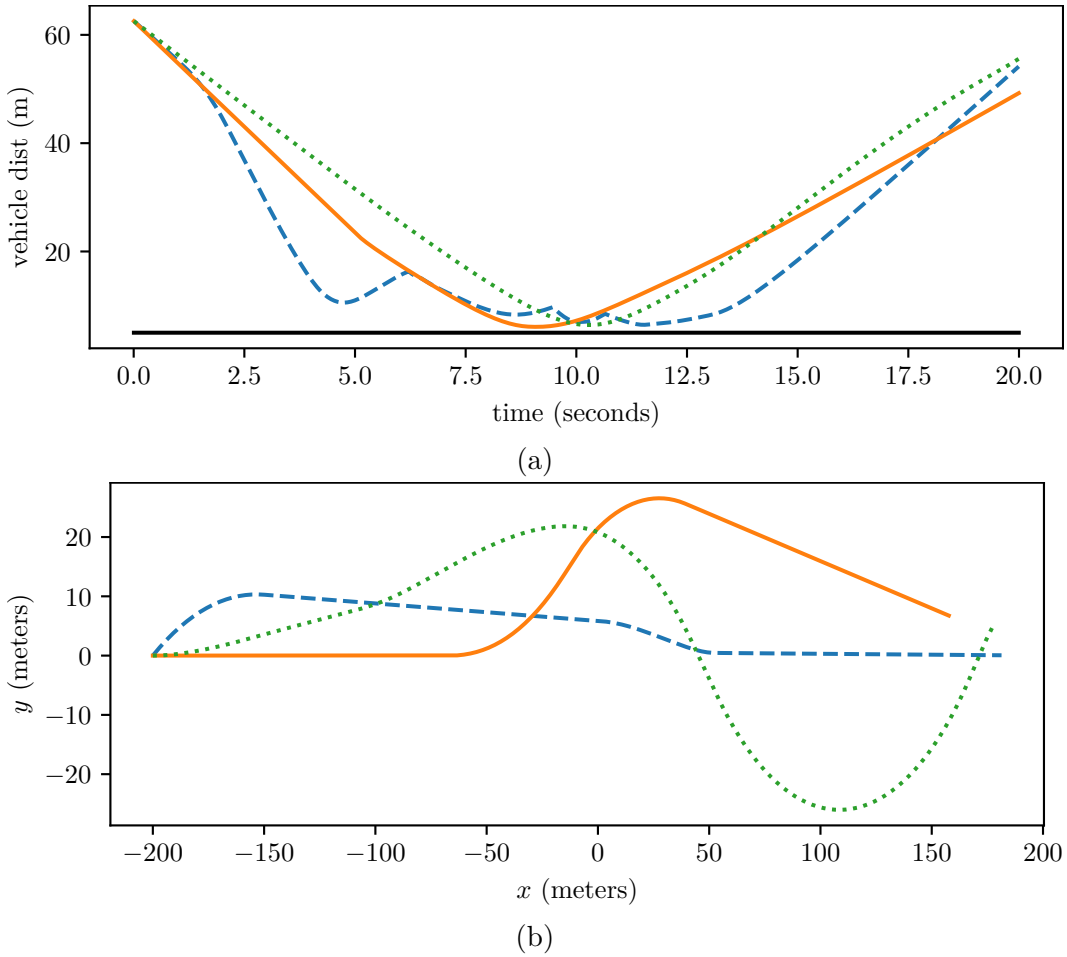


Figure 4.4: Outputs for the scenario with 20 fixed-wing vehicles. The blue dashed and orange solid lines are the output of the scenario where h is constructed from $\gamma_{straight}$ and γ_{turn}, respectively. The green dotted line is the output from using a navigation function. Vehicle 1 is plotted as a representative output since all 20 vehicles cannot be shown on the same plot. In (c), the minimum distance between any two vehicles is shown to be above D_s. (d) is the path taken by vehicle 1. Note that the behavior is significantly different when constructing h with γ_{turn} and $\gamma_{straight}$. The navigation function starts avoiding a collision well in advance of when using h constructed from γ_{turn}.

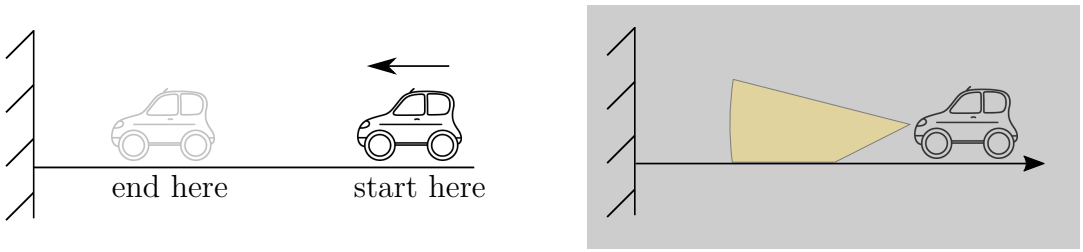
CHAPTER 5

SAFETY WITH LIMITED RANGE SENSORS

5.1 Introduction

Chapters 3 and 4 relaxed two assumptions from Chapter 2, namely that there was only one safety objective to be satisfied at all times and that the vehicles could communicate high numbers of messages, respectively. In this chapter we relax another assumption, namely that the vehicles can sense each other at all times. Recall the discussion at the beginning of the last chapter, where we noted that to calculate a safe control value, the vehicles must know the shared evading maneuver, the safety objectives, the nominal control value, and the state. The last chapter removed the need to know the nominal control value of other vehicles. This chapter removes the need to always know the state. In particular, we consider safety when the state is not known at all times and find that in some cases we can still ensure the system stays safe.

In the prior chapters, we have shown how to ensure a system of k UAVs maintain safe distances for all time while taking into account dynamics constraints. However, the discussion did not consider limited range sensing. Thus, in this chapter we relax this limitation with the following contributions. First, we show that the barrier functions do not necessarily guarantee safety when the UAVs are subject to limited range sensing. Second, we introduce a method for constructing a new barrier function that accommodates limited sensing range from a previously existing barrier function that may not necessarily accommodate limited range sensing. Finally, we conduct an experiment consisting of a scenario of 20 fixed wing aircraft, where because of the proposed algorithm, the vehicles are able to maintain safe distances from each other



(a) (b)

Figure 5.1: (left) When there is infinite sensing the car knows when to start decelerating in order to avoid the wall. (right) When there is limited sensing the car might not sense the wall until it is too late.

even though the vehicles are subject to limited range sensing.

5.2 Motivation

Recall the car example of Figure 1.1 where a car starts some distance from the wall. Given its current position and velocity, we specify an evasive maneuver to maximally accelerate away from the wall, forward propagate under this hypothetical control action, and let the barrier function be the closest distance the vehicle gets to the wall under this forward propagation. Consider then when the car has a limited sensing range due to lighting conditions. If the sensing range becomes too small to stop after initially sensing the wall then collision avoidance can no longer be guaranteed (see Figure 5.1). We further motivate this idea with specific examples from collision avoidance for fixed-wing aircraft.

We assume there is a sensor modeled via a set $S \subset \mathcal{D}$ such that, if the system state x is such that $x \in S$, then x is completely known to both vehicles, whereas if $x \notin S$, then all that is known is that $x \notin S$. In the case of UAV collision avoidance where each UAV is equipped with an omnidirectional sensor with range R, $S = \{x \in \mathcal{D} : d_{1,2}(x) \leq R^2\}$. In this section we present two motivating examples to illustrate two distinct issues that can arise when using barrier functions in the presence of limited range sensing. In both cases, the critical problem is that $K(x)$ cannot be calculated

for all $x \in \mathcal{D}$ because $S \subset \mathcal{D}$.

The first issue that limited range sensing introduces is that that safety can no longer be guaranteed. In particular, we construct a scenario where $h(x(0)) \geq 0$ and, because $K(x)$ cannot be calculated, there is a future time for which $h(x(t)) < 0$. In other words, we can have a continuously differentiable barrier function h that satisfies (1.5) but still not be able to guarantee safety. The second issue we examine is that there can be discontinuities in actuator commands even though h is continuously differentiable. This can cause alarm or discomfort for systems designed to ensure safety of human passengers (e.g., cruise control [4]). In the following examples, consider two UAVs equipped with omnidirectional sensors (e.g. radar) of radius R, with dynamics governed by a nominal controller $\hat{u}(x)$. See Figure 5.2 for illustrations.

Example 3. A Barrier Function Without a Safety Guarantee. Suppose the two vehicles start at $x_1(0) = \begin{bmatrix} r_1 + R/2 & r_1 & -\pi/2 & 0 \end{bmatrix}^T$, $x_2(0) = \begin{bmatrix} -r_2 - R/2 & r_2 & -\pi/2 & 0 \end{bmatrix}^T$, respectively, where vehicle 1 has a nominal control input of $\hat{u}_1 = \begin{bmatrix} v_1 & -\omega \end{bmatrix}^T$ and vehicle 2 has a nominal control input of $\hat{u}_2 = \begin{bmatrix} v_2 & \omega \end{bmatrix}^T$ so that they both follow a circular trajectory with radius $r_1 = v_1/\omega$ and r_2/ω, respectively (see Figure 5.2a). Then $h_{straight}(x(0)) = (r_1 + r_2 + R) - D_s \geq 0$ as long as $R \geq \max(0, D_s - r_1 - r_2)$ so the vehicles start safe according to $h_{straight}$. Further, note that because the vehicles cannot sense each other, $K(x)$ cannot be calculated. This is because to calculate $K(x)$, the values of $L_f h(x)$, $L_g h(x)$, and $h(x)$ are required. Because $K(x)$ cannot be calculated, there is no means to ensure that the control input applied to the vehicle will be in $K(x)$. In particular, it means that it is unknown whether the nominal control input $\hat{u}$ is in $K(x)$ and a design decision must be employed for what actuation input to apply to the vehicles. If the design decision is, for instance, to apply the nominal controller to the vehicles then the vehicles will reach $\begin{bmatrix} R/2 & 0 & \pi & 0 \end{bmatrix}^T$ and $\begin{bmatrix} -R/2 & 0 & 0 & 0 \end{bmatrix}^T$, respectively. Once the vehicles have reached this state, $h_{straight}(x) = -D_s$. This

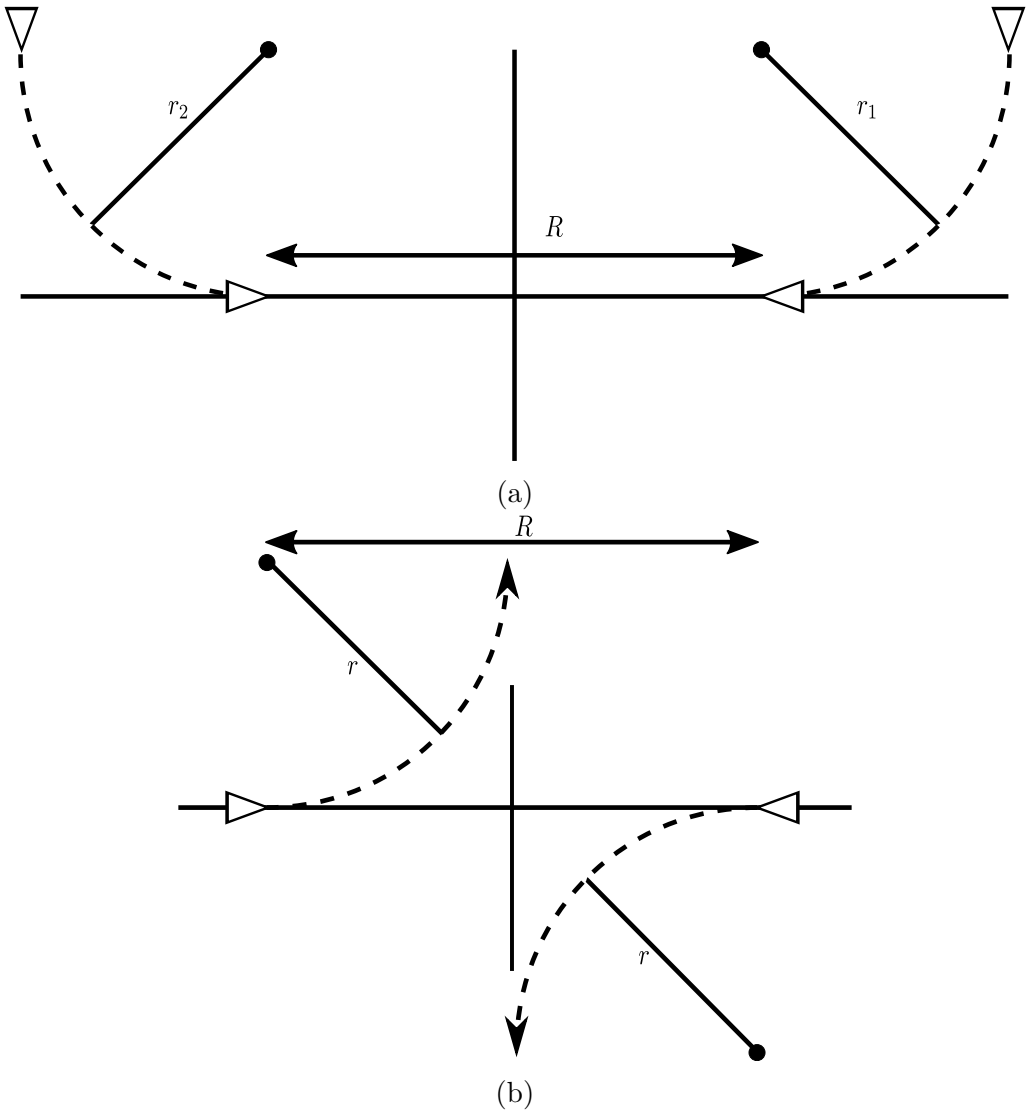


Figure 5.2: Two examples where limited range sensing creates issues when applying barrier functions to fixed-wing aircraft collision avoidance. In (a), the vehicles start so that $h_{straight}(x) = (r_1 + r_2 + R) - D_s \geq 0$ but because the vehicles cannot sense each other, achieve a configuration where $h_{straight}(x) = -D_s$. In (b), the vehicles travel along the x-axis until they sense each other at a distance of R apart, at which point the safe control implied by h_{turn} requires high turn speed. Adapted with permission from [67] ©2021 IEEE.

means that the vehicles started in a state x such that $h(x(0)) \geq 0$ but there exists a later time t such that $h(x(t)) < 0$. This is because $K(x)$ cannot be calculated for $x \notin S$ so the control input applied to the aircraft does not always satisfy (1.5). In other words, because $K(x)$ cannot be calculated for all $x \in \mathcal{D}$, Theorem 1 cannot be used to guarantee safety.

Example 4. Loss of Smoothness. For h_{turn} let γ_{turn} be specified with $v = v_{min}$ and $\omega = \omega_{max}$ in (2.9). Let the two aircraft have sensor radius $R = (D_s + 2r)\cos(\eta) + 4\delta$ where $r = v_{min}/\omega_{max}$ and $\eta = \arcsin(r/(r + D_s/2))$. As in Figure 5.2b, let the vehicles have initial positions of

$$\begin{bmatrix} (D_s/2 + r)\cos(\eta) + 2\delta + \epsilon & 0 & -\pi & 0 \end{bmatrix}^T$$

and

$$\begin{bmatrix} -(D_s/2 + r)\cos(\eta) - 2\delta - \epsilon & 0 & 0 & 0 \end{bmatrix}^T,$$

respectively, where $\epsilon > 0$. See Figure 5.3 for the geometric setup. Further, let each aircraft have a nominal trajectory that continues toward the origin. Because the aircraft cannot sense each other, $K(x)$ cannot be calculated. This means that there will be no collision avoidance override so the applied actuator command will be equal to the nominal controller command of $\hat{u}_i(x) = \begin{bmatrix} v_{max} & 0 \end{bmatrix}^T$ until the vehicles reach states

$$\begin{bmatrix} (D_s/2 + r)\cos(\eta) + 2\delta & 0 & -\pi & 0 \end{bmatrix}^T$$

and

$$\begin{bmatrix} -(D_s/2 + r)\cos(\eta) - 2\delta & 0 & 0 & 0 \end{bmatrix}^T,$$

respectively. At this point, the vehicles can sense each other and the constraints (1.5) in the QP (1.10) can be calculated, resulting in a discontinuity in the constraint in the QP (1.10).

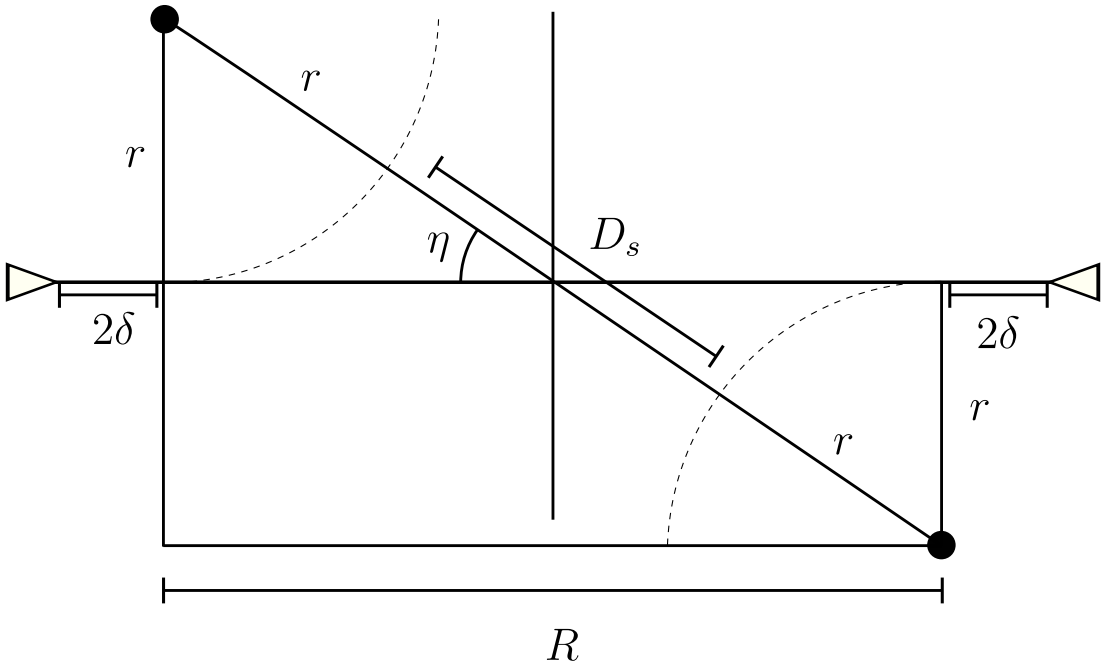


Figure 5.3: The geometric setup for the chosen variables for Example 4.

5.3 Constructing a Barrier Function for Safety Guarantees Despite Limited Range Sensing Restrictions

In Section 5.2 we saw that limited range sensing can lead to practical issues including the loss of safety guarantees even when a ZCBF, h, exists for the system. This means that UAVs may collide with each other when they are equipped with limited range sensors. When limited sensing is not taken into account in the design of a ZCBF h, the problem is that values of h cannot be evaluated for all $x \in \mathcal{D}$, as required by Definition 1, and so h cannot be used to guarantee safety. In this section we provide a solution to this issue.

Definition 3. For a given ZCBF h and a sensor with a sensor set S, h is sensor compatible if h is a positive constant for all $x \notin S$.

Remark 6. A sensor compatible ZCBF h must be positive outside S since otherwise this would imply for $x \notin S$ that (1.5) becomes $\alpha(h(x)) < 0$ so (1.5) does not hold for any $u \in U$.

Importantly, implementing a safety overriding controller requires an exact calculation of the ZCBF constraint (1.5) only if $K(x) \neq U$. When $K(x) = U$, there is no need to calculate $h(x)$ or its derivatives because any $u \in U$ is already known to be safe. Because of the additional structure on a sensor compatible ZCBF, we can relax the need to check $u \in K(x)$ for all $x \in \mathcal{D}$ in Theorem 1 because it is already known that $u \in K(x)$ for all u when $x \notin S$, as is made precise in the following Corollary.

Corollary 2. Suppose h is a sensor compatible ZCBF. Then any Lipschitz continuous controller $u : \mathcal{D} \to U$ such that $u(x) \in K(x)$ for all $x \in S$ will render the set $\mathcal{C}$ forward invariant.

Proof. By assumption $u(x) \in K(x)$ for all $x \in S$. Suppose then that $x \notin S$ so that $h(x)$ is a positive constant. Then $K(x) = U$ since $L_f h(x) + L_g h(x)u + \alpha(h(x)) = \alpha(h(x)) > 0$ is satisfied for all $u \in U$. Hence $u(x) \in K(x)$ for all $x \in \mathcal{D}$ so the assumptions of Theorem 1 are satisfied. □

Remark 7. The difference between Theorem 1 and Corollary 2 is that the condition $u(x) \in K(x)$ only needs to be the case for $x \in S$ rather than $x \in \mathcal{D}$ due to the extra structure on a sensor compatible ZCBF. This is an important distinction because when there are sensing limitations, it may not be possible to calculate $K(x)$ for all $x \in \mathcal{D}$.

Remark 8. For an arbitrary sensor, neither $h_{straight}$ nor h_{turn} are necessarily sensor compatible. To see this for $h_{straight}$, let $R > 0$, $x_1 = \begin{bmatrix} R+\epsilon & D_s & \pi & 0 \end{bmatrix}^T$, and $x_2 = \begin{bmatrix} -R+\epsilon & 0 & 0 & 0 \end{bmatrix}^T$. Then $x \notin S$ for $\epsilon > 0$ and in this case $h(x) = 0$. However, if $x_2 = \begin{bmatrix} -R+\epsilon & -D_s & 0 & 0 \end{bmatrix}^T$, then $x \notin S$ but $h(x) = D_s$. Then $h_{straight}$ is not sensor compatible because $h(x)$ is not constant for all $x \notin S$. A similar calculation can be done to show h_{turn} is not sensor compatible. Hence, we cannot always apply Corollary 2 to $h_{straight}$ or h_{turn} when there is limited range sensing.

Consider now some ZCBF h that is not necessarily sensor compatible. We now show how to create $\widetilde{h}$ from h so that $\widetilde{h}$ is sensor compatible even though this is not the case for h. This allows us to be able to apply Corollary 2 to $\widetilde{h}$. However, it is not always possible to create $\widetilde{h}$ from h so that $\widetilde{h}$ is sensor compatible and we therefore consider two cases. First, although h_{turn} is not necessarily sensor compatible for an arbitrary sensor, we give sufficient conditions to construct a ZCBF to be sensor compatible. Second, we show how to verify when it is impossible to construct a sensor compatible ZCBF using the proposed method. We show this is the case for $h_{straight}$.

To construct $\widetilde{h}$, we first introduce an interpolation function to ensure that $\widetilde{h}$ is continuously differentiable, as required by the definition of a ZCBF. Let $\xi > 0$, $0 < \beta < 1$, and ψ be a continuously differentiable, non-decreasing, real valued function on an open set including $[\beta\xi, \xi]$ chosen to satisfy

$$\begin{aligned} \psi(\beta\xi) &= \beta\xi \\ \psi'(\beta\xi) &= 1 \\ \psi'(\xi) &= 0. \end{aligned} \tag{5.1}$$

Example 5. An example of such a function ψ can be found by fitting a quadratic function. Let $\psi(\eta) = c_1\eta^2 + c_2\eta + c_3$. Then (5.1) can be solved for $c_1 = \frac{-1}{2\xi(1-\beta)}$, $c_2 = -2\xi c_1$, and $c_3 = \beta\xi - c_1(\beta\xi)^2 - c_2\beta\xi$.

We now define $\widetilde{h}$ as follows

$$\widetilde{h}(x) = \begin{cases} h(x) & h(x) \leq \beta\xi \\ \psi(h(x)) & \beta\xi < h(x) < \xi \\ \psi(\xi) & \xi \leq h(x) \end{cases} \tag{5.2}$$

where we let $B_\xi = \{x \in \mathcal{D} \,:\, h(x) \leq \xi\}$ be a sub-level set of h (see Fig 5.4), where

ξ denotes the maximum value of h for which the safety constraint affects the value of $\widetilde{h}$. B_ξ is the set of states where the safety constraint affects the value of $\widetilde{h}$. β is a mixing term for states where $\beta\xi < h(x) < \xi$ and exists to ensure the differentiability of $\widetilde{h}$.

Remark 9. With this setup, we can take the following steps to show when a ZCBF $\widetilde{h}$ is sensor compatible. First, the system designer chooses ξ which determines B_ξ. Second, the system designer determines if $B_\xi \subseteq S$. In other words, $\xi > 0$ must be chosen with the sensing range in mind in order to verify that $B_\xi \subseteq S$. The second step verifies that a sensor exists so that any value of $h(x)$ such that $h(x) < \xi$ can be calculated. Since $\widetilde{h}$ does not require knowledge of x for $h(x) \geq \xi$, $\widetilde{h}(x)$ can be calculated for all $x \in \mathcal{D}$. We prove this intuition below. Note that when multiple ξ satisfy the above steps, it may be preferable to select larger ξ because $h(x) = \widetilde{h}(x)$ for all $x \in \mathcal{D}$ such that $h(x) \leq \beta\xi$.

Lemma 1. Assume h defined in (2.3) is a ZCBF where γ is locally Lipschitz. Let $\widetilde{h}$ be defined as in (5.2). Then $\widetilde{h}$ is a ZCBF on $\mathcal{D}$.

Proof. Note that because of how ψ is defined and because h is a continuously differentiable function, that $\widetilde{h}$ is a continuously differentiable function. Also note that for $\beta\xi < h(x) < \xi$, $\psi(h(x)) > 0$ since ψ is a non-decreasing function that is positive at $\beta\xi$.

To show that $\widetilde{h}$ satisfies (1.5), let $x \in \mathcal{D}$. We consider three cases. First, if $h(x) \leq \beta\xi$ then the inequality (1.5) holds for $\widetilde{h}(x)$ because $h(x)$ is a ZCBF. If $h(x) \geq \xi$ then (1.5) for $\widetilde{h}(x)$ becomes $\alpha(\psi(\xi)) \geq 0$ which is true for all $u \in U$ because $\psi(\xi) > 0$ and α is a class $\mathcal{K}$ function. Finally, suppose $\beta\xi < h(x) < \xi$ and note that because ψ

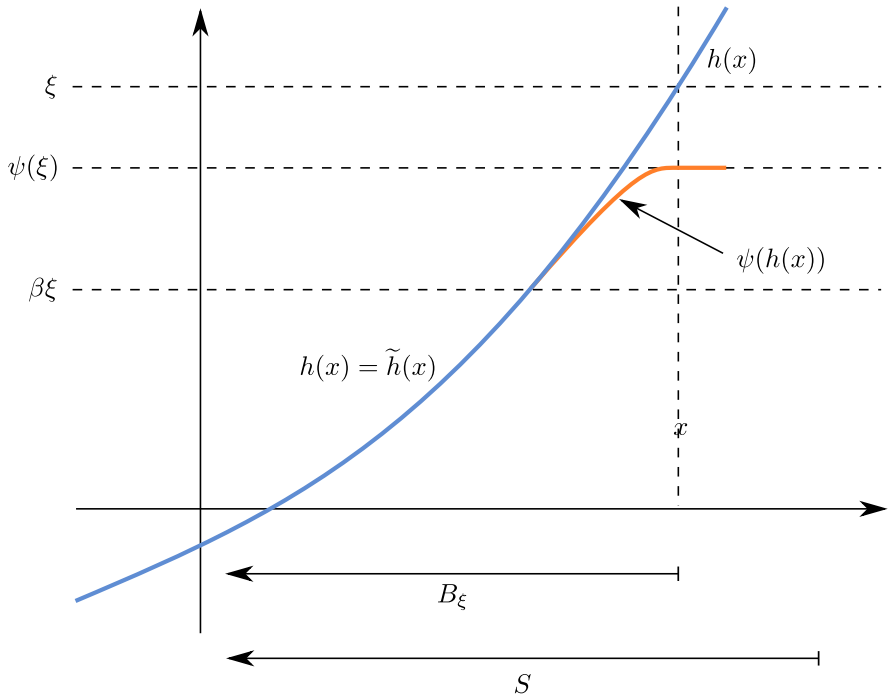


Figure 5.4: A graphical view of a hypothetical barrier function h (blue) along with $\tilde{h}$ (orange) of a one-dimensional system. Given h, the system designer's choice of ξ and β defines $\tilde{h}$ and B_ξ. According to Theorem 5 when the system designer can identify a $\xi > 0$ that defines B_ξ where $B_\xi \subseteq S$, $\widetilde{h}$ is a ZCBF compatible with a sensor s. Adapted with permission from [67] ©2021 IEEE.

is non-decreasing that $\frac{\partial \psi(h(x))}{\partial h(x)} \geq 0$. Then

$$\begin{aligned} & L_f \widetilde{h}(x) + L_g \widetilde{h}(x)\gamma(x) + \alpha(\widetilde{h}(x)) \\ & \quad = \frac{\partial \psi(h(x))}{\partial h(x)} \left(L_f h(x) + L_g h(x)\gamma(x)\right) + \alpha(\psi(h(x))) \\ & \quad \geq 0. \end{aligned}$$

The first line uses the chain rule. The second line uses the fact that $\frac{\partial \psi(h(x))}{\partial h(x)} \geq 0$ and that $L_f h(x) + L_g h(x)\gamma(x) \geq 0$, as was established in the proof of Theorem 2. Finally, note that $\alpha(\psi(h(x))) \geq 0$ because $\psi(h(x)) > 0$ and α is a class $\mathcal{K}$ function. Then $\widetilde{h}$ is a ZCBF. □

Theorem 5. For a given sensor S, assume h defined in (2.3) is a ZCBF where γ is locally Lipschitz and there exists a $\xi > 0$ such that $B_\xi \subseteq S$. Then $\widetilde{h}$ defined in (5.2) is a sensor compatible ZCBF.

Proof. $\widetilde{h}$ is a ZCBF by Lemma 1. Suppose $x \notin S$. Then because $B_\xi \subseteq S$, $h(x) > \xi$ so $\widetilde{h}(x) = \psi(\xi)$. Then $\widetilde{h}$ is a positive constant for all $x \notin S$ and is sensor compatible. □

Theorem 5 is the justification of the steps listed in Remark 9. Combined with Corollary 2, Theorem 5 states how the forward invariance of a set $\mathcal{C}$ can be guaranteed even though there is limited range sensing. However, it is predicated on finding a $\xi > 0$ that defines a sublevel set B_ξ for which $B_\xi \subseteq S$. We now give an example of such a case for fixed wing collision avoidance where each aircraft is equipped with an omnidirectional sensor with a given range R.

Example 6. It was shown in Remark 8 that h_{turn} is not necessarily sensor compatible for an arbitrary sensor. We now use Theorem 5 to define sensing requirements so that we can create $\widetilde{h}$ from h_{turn} so that $\widetilde{h}$ is sensor compatible. For h_{turn}, the trajectory defined in (2.4) is a circle for each aircraft with radius $r_1 = \sigma v/\omega$ and $r_2 = v/\omega$, respectively. Let $\Delta x(t) = p_{1,x}(t) - p_{2,x}(t)$, and $\Delta y(t) = p_{1,y}(t) - p_{2,y}(t)$, so that the

vehicles start at a planar distance of $(\Delta x^2(t) + \Delta y^2(t))^{1/2}$ from each other. Assuming the planar distance between the vehicles, $(\Delta x^2(t) + \Delta y^2(t))^{1/2}$, is greater than $2(r_1 + r_2)$, the closest the distance can be along the trajectory (2.4) is $d_{1,2}(x(0))^{1/2} - 2r_1 - 2r_2$.

Assume each vehicle has an omnidirectional sensor with range R large enough so that

$$\left((R - 2r_1 - 2r_2)^2 - 4\delta\right)^{1/2} - D_s > 0. \tag{5.3}$$

Equation (5.3) implies S for this example. Having defined the sensing limitations for this problem, we now follow the steps in Remark 9 to show that $\widetilde{h}$ is a sensor compatible ZCBF. First, we choose ξ so that we can prove $B_\xi \subseteq S$, namely

$$\left((R - 2r_1 - 2r_2)^2 - 4\delta\right)^{1/2} - D_s = \xi > 0. \tag{5.4}$$

Second, we show that $B_\xi \subseteq S$. Suppose $x(t) \notin S$ so that $d_{1,2}(x(t)) > R^2$. Then because the trajectories of each vehicle is a circle in (2.4),

$$\begin{aligned} h(x(t)) &= \inf_{\tau \in [0,\infty)} \rho(t + \tau) \\ &\geq ((d_{1,2}(x(0))^{1/2} - 2r_1 - 2r_2)^2 - 4\delta)^{1/2} - D_s \\ &> ((R - 2r_1 - 2r_2)^2 - 4\delta)^{1/2} - D_s \\ &= \xi \\ &> 0. \end{aligned}$$

Then $x(t) \notin B_\xi$. Then $B_\xi \subseteq S$. In other words, given a sensor of radius R, we can choose ξ according to (5.4) and use $\widetilde{h}$ defined in (5.2) to ensure safe operations between two fixed wing aircraft.

While Example 6 showed how to use h_{turn} with Theorem 5, the same cannot be done for $h_{straight}$ (see Figure 5.5).

Corollary 3. Assume h defined in (2.3) is a ZCBF where γ is locally Lipschitz. Suppose

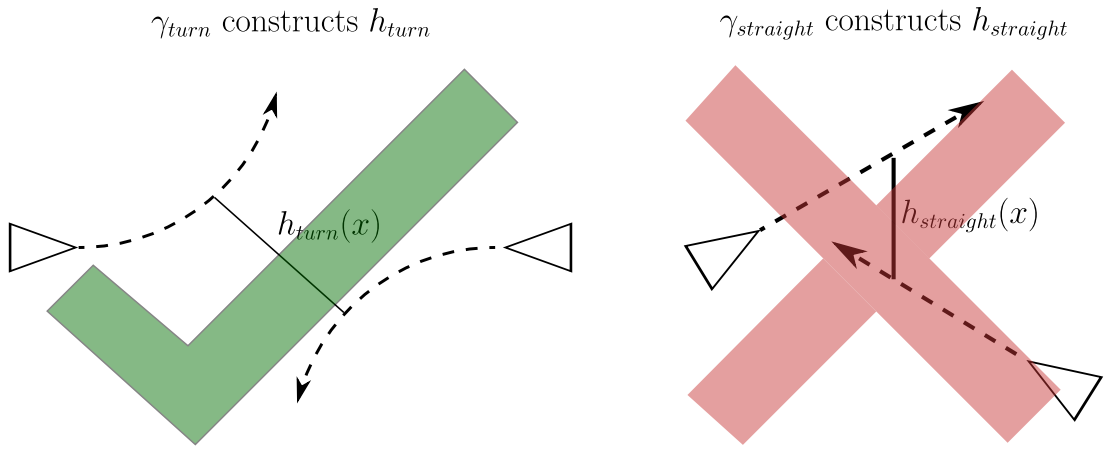


Figure 5.5: Theorem 5 can be used to calculate specify precise sensing range requirements to ensure safety guarantees but there is no finite range sensor for which the same can be done for $h_{straight}$.

there exists an $x \in \mathcal{D}$ such that $h(x) < 0$ and $x \notin S$. Then for all $\xi > 0$, $B_\xi \not\subseteq S$.

Proof. Note that for the given x, $x \in B_\xi$ for any $\xi > 0$ since $h(x) < 0$. Then $x \in B_\xi$ but $x \notin S$. □

Remark 10. We now use Corollary 3 to show $h_{straight}$ cannot be used with Theorem 5 to guarantee safety. Let $x_1(0) = \begin{bmatrix} -R/2 - \epsilon & 0 & 0 & 0 \end{bmatrix}^T$ $x_2(0) = \begin{bmatrix} R/2 + \epsilon & 0 & \pi & 0 \end{bmatrix}^T$, the sensing radius be R, and $\epsilon > 0$, Then $h(x) = -D_s$ and $x \notin S$ since the vehicles are further than R apart.

5.4 An Interpretation of $\widetilde{h}$ As a More Permissive ZCBF Than h

Consider again Theorem 5, which gives sufficient conditions to guarantee safety even though the state cannot always be sensed. The key idea that leads to Theorem 5 is the construction of a barrier function $\widetilde{h}$ that is constant for the set of states that cannot be sensed. In other words, the coefficient on u in equation (1.5) is zero so any u satisfying the actuator constraints also satisfies the safety constraint. This means that if the nominal controller is within the actuator constraints that there will be no safety override to alter the nominal control value. What we should expect in this case

is that the state can approach the safety boundary more quickly than if there were not any sensing restrictions. In this section we make this intuition precise by showing the conditions for which using $\widetilde{h}$ as the barrier function is at least as permissive as the original barrier function h.

Consider a ZCBF h that is not necessarily sensor compatible but for which it is possible to construct $\widetilde{h}$ so that $\widetilde{h}$ is a sensor compatible ZCBF. In this section we characterize how $\widetilde{h}$ is more permissive than h. For notational convenience we denote $\nabla_h \psi(h(x)) = \frac{\partial \psi(h(x))}{\partial h(x)}$ and assume the following.

Assumption 2. Assume on $(\beta\xi, \xi)$, α is continuously differentiable. Further, assume the derivative of α is non-increasing on $(\beta\xi, \xi)$.

Remark 11. The assumptions on α can be satisfied for any α that is linear on the region $(\beta\xi, \xi)$.

Assumption 3. Assume the domain of ψ is extended by letting ψ be the identity function for inputs $h(x) < \beta\xi$. Further, assume the first derivative of ψ is strictly positive but non-increasing on $(\beta\xi, \xi)$ and the second derivative of ψ is negative on $(\beta\xi, \xi)$.

Remark 12. Note that because the first derivative of ψ is strictly positive for $h(x) < \xi$, $(\nabla_h \psi(h(x)))^{-1}$ is well defined on $h(x) < \xi$.

Remark 13. The ψ discussed in Example 5 satisfies Assumption 3 by letting $\psi(h(x)) = h(x)$ for $h(x) \leq \beta\xi$.

We begin by expanding the barrier condition (1) for $\widetilde{h}$ in terms of h. This will then be used to show the conditions such that $K(x) \subseteq \widetilde{K}(x)$ where $\widetilde{K}(x)$ is the admissible control space of $\widetilde{h}$ at x. Under Assumption 3 and noting Remark 12, for $h(x) < \xi$ we

have for $x \in \mathcal{D}$ that

$$\begin{aligned}
&L_f\widetilde{h}(x) + L_g\widetilde{h}(x)u + \alpha(\widetilde{h}(x)) \\
&\quad = \nabla_h\psi(h(x))\left(L_fh(x) + L_gh(x)u\right) + \alpha(\psi(h(x))) \\
&\quad = \nabla_h\psi(h(x))\Big(L_fh(x) + L_gh(x)u + \\
&\qquad\qquad (\nabla_h\psi(h(x)))^{-1}\alpha(\psi(h(x)))\Big).
\end{aligned}$$

Then because $\nabla_h\psi(h(x)) > 0$ and letting

$$\alpha_2(h(x)) = (\nabla_h\psi(h(x)))^{-1}\alpha(\psi(h(x))), \tag{5.5}$$

we have, letting sgn be the sign of the expression,

$$\begin{aligned}
&\operatorname{sgn}(L_f\widetilde{h}(x) + L_g\widetilde{h}(x)u + \alpha(\widetilde{h}(x))) \\
&\quad = \operatorname{sgn}(L_fh(x) + L_gh(x)u + \alpha_2(h(x))).
\end{aligned} \tag{5.6}$$

Lemma 2. Suppose h is a ZCBF, let $\widetilde{h}$ be as defined in (5.2), let Assumptions 2 and 3 hold, and let α_2 be as defined in (5.5). If $h(x) < \xi$ then $\alpha_2(h(x)) \geq \alpha(h(x))$.

Proof. Suppose $h(x) \leq \beta\xi$. Then because $\psi(h(x)) = h(x)$, $\alpha_2(h(x)) = \alpha(h(x))$. Suppose now that $\beta\xi < h(x) < \xi$. We prove $\alpha_2(h(x)) \geq \alpha(h(x))$ with the comparison lemma [68].

It has already been shown that $\alpha_2(h(x)) = \alpha(h(x))$ at $h(x) = \beta\xi$. We now show

that $\nabla_h \alpha_2(h(x)) \geq \nabla_h \alpha(h(x))$ for $h(x) \in (\beta\xi, \xi)$. By the chain rule,

$$\begin{aligned}
&\nabla_h \alpha_2(h(x)) \\
&\quad = -\left(\frac{1}{\nabla_h \psi(h(x))}\right)^2 \nabla_h^2 \psi(h(x)) \alpha(\psi(h(x)) \\
&\qquad + \left(\frac{1}{\nabla_h \psi(h(x))}\right) \nabla_\psi \alpha(\psi(h(x))) \nabla_h \psi(h(x)) \\
&\quad \geq \nabla_\psi \alpha(\psi(h(x))).
\end{aligned}$$

The inequality holds because the second derivative of ψ is negative and $\alpha(\psi(h(x)) \geq 0$ for $h(x) \in (\beta\xi, \xi)$. We must now show that $\nabla_\psi \alpha(\psi(h(x))) \geq \nabla_h \alpha(h(x))$ to conclude that $\alpha_2(h(x)) \geq \alpha(h(x))$ for $h(x) \in (\beta\xi, \xi)$.

Because $\psi(h(x)) = h(x)$ for $h(x) = \beta\xi$, the first derivative of ψ is 1 at $h(x) = \beta\xi$ and the first derivative is non-increasing on $h(x) \in (\beta\xi, \xi)$, so $\psi(h(x)) \leq h(x)$ for $h(x) \in (\beta\xi, \xi)$. Then because the derivative of α is non-increasing for $h(x) \in (\beta\xi, \xi)$, $\nabla_\psi \alpha(\psi(h(x)) \geq \nabla_h \alpha(h(x))$.

□

Remark 14. Note that α_2 is a class $\mathcal{K}$ function. By definition, $\alpha_2(0) = 0$ and is strictly increasing on $(0, \beta\xi)$. To see that α_2 is strictly increasing on $(\beta\xi, \xi)$, note that it has already been proven that $\nabla_h \alpha_2(h(x)) \geq \nabla_\psi \alpha(\psi(h(x))) \geq \nabla_h \alpha(h(x))$. Further $\nabla_h \alpha(h(x)) > 0$ since α is a class $\mathcal{K}$ function. Then $\nabla_h \alpha_2(h(x)) > 0$.

Theorem 6. Suppose h is a ZCBF, assume $\widetilde{h}$ be as defined in (5.2) is sensor compatible, and let Assumptions 2 and 3 hold. Then $K(x) \subseteq \widetilde{K}(x)$ for all $x \in \mathcal{D}$.

Proof. Suppose x is such that $h(x) < \xi$ and $u \in K(x)$. Then $L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0$. Then since $\alpha_2(h(x)) \geq \alpha(h(x))$ from Lemma 2 we have $L_f h(x) + L_f h(x) u + \alpha_2(h(x)) \geq 0$. Then from (5.6), $L_f \widetilde{h}(x) + L_g \widetilde{h}(x) u + \alpha(\widetilde{h}(x)) \geq 0$. Then $u \in \widetilde{K}(x)$.

Suppose x is such that $h(x) \geq \xi$ and $u \in K(x)$. Then because $u \in U$, $u \in \widetilde{K}(x)$ since $\widetilde{K}(x) = U$. $\square$

Remark 15. In particular, Theorem 6 gives the conditions under which any u satisfying the QP (1.10) using h will be satisfied in a QP (1.10) when using $\widetilde{h}$.

Theorem 6 shows the conditions under which $\widetilde{h}$ is at least as permissive as the original barrier function h. This means a safety override based on $\widetilde{h}$ will be less invasive than a barrier function based on h. However, it is important to ask whether this is something a system designer actually wants. In particular, we generally want to take small overriding actions when the state is not yet close to the safety boundary in order in case there is noise in the modeling or sensors. This is the role of α in (1.5).

In other words, when using h with unlimited sensing, the safety overriding controller will start to take action when the system is further from the safety boundary. When there is a loss of sensing outside some radius, the state can travel arbitrarily quickly (within actuator limits) towards the boundary. Although safety can still be guaranteed in this case, it may involve approaching the boundary more quickly than desired. Incorporating longer range sensors directly solves this issue but if this is not possible, this discussion implies introducing additional conservatism elsewhere may be desirable. Examples of where additional conservatism can be introduced are in α, the evading maneuver, or adding artificial actuator limits. For instance, instead of using cubic α, a linear alpha with a small coefficient may be preferable. Similarly, as discussed in Chapter 2 it may be preferable to choose an evasive maneuver well within the actuator limits so there is significant room to deviate from the modeled evasive maneuver. This was shown to be important for a ZCBF to satisfy (1.5) outside of the safe set in the proof of Theorem 2. Finally, by adding artificial actuator limits (e.g., limiting the actual speed of the aircraft to something much smaller than v_{max}) can also reduce how quickly the vehicles approach the safety boundary.

5.5 Simulation Experiments

In this section we conduct a simulation experiment with SCRIMMAGE [59]. We consider two vehicles with initial states of $\begin{bmatrix} -200 & 0 & 0 & 0 \end{bmatrix}^T$ and $\begin{bmatrix} 200 & 0 & \pi & 0 \end{bmatrix}^T$ with goal positions of $\begin{bmatrix} 200 & 0 & 0 \end{bmatrix}^T$ and $\begin{bmatrix} -200 & 0 & 0 \end{bmatrix}^T$, respectively.

We use h_{turn}, letting $v = 0.9v_{min} + 0.1v_{max}$ and $\omega = 0.9\omega_{max}$ in (2.9) where $v_{min} = 15$ meters/second, $v_{max} = 25$ meters/second, $\delta = 0.01$ meters^2, and $\omega_{max} = 13$ degrees/second. The choice of ω_{max} results from assuming a 30 degree max bank angle while traveling at v_{max} and using a constant rate turn formula, as in [61].

In the first experiment, we examine the effect of sensing range on the resulting closest distance the vehicles experience during the simulation. In Example 6 we showed how to apply the steps described in Remark 9 by choosing ξ so that $B_\xi \subseteq S$ to conclude that that $\widetilde{h}$ is sensor compatible. The conclusion required that the sensing range was above a threshold in (5.3). Using the parameters of this experiment, equation (5.3) implies $R > 318.4$. Because the inequality is strict, we start the experiment with $R = 319$. As shown in Figure 5.6, provided the sensing range is above the threshold calculated in (5.3), the vehicles are able to maintain safe distances throughout the simulation. Further, as the sensing range approaches the limit predicted by (5.3), the minimum distance between the vehicles during the simulation approaches D_s.

In the second experiment we repeat the experiment of Chapter 4 where 20 vehicles are applying a barrier function and are positioned around a circle with a nominal controller that cause the vehicles to arrive at the origin at the same time. The difference in this experiment from Chapter 4 is that we include a limited sensing range of 350 for each vehicle and start the vehicles 1250 feet from the origin so they start the scenario without being able to sense each other. A screenshot of the initial conditions and evasive maneuver are shown in Figure 5.9 and a video simulation

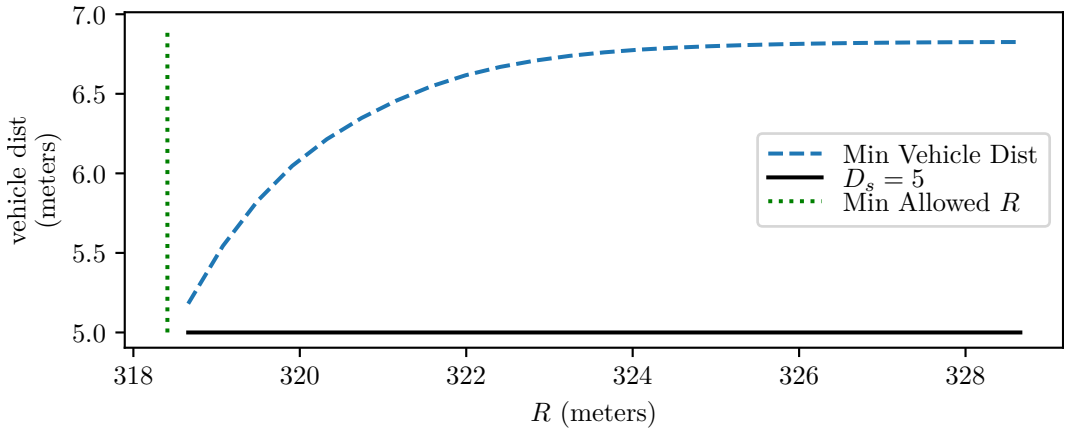
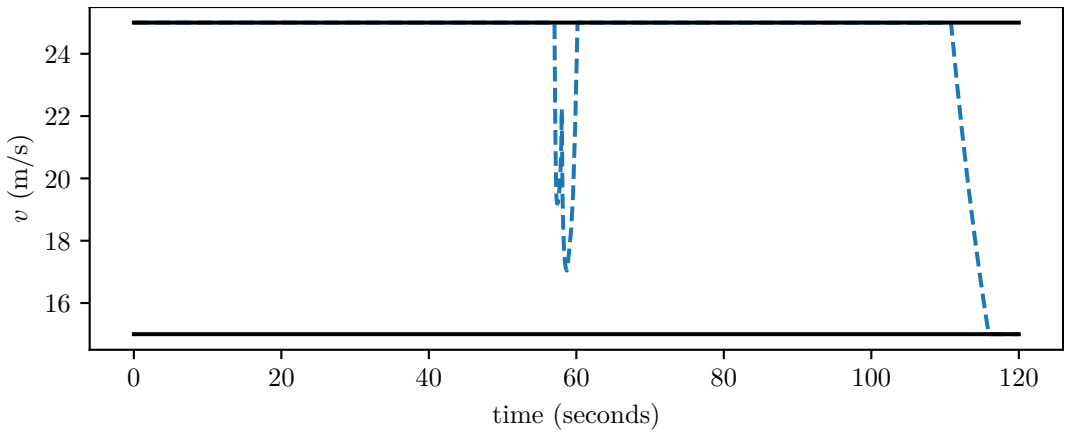


Figure 5.6: The minimum vehicle distance vs sensing range. The green dashed line is the minimum sensing range using (5.3). Note that for all sensing ranges above the minimum sensing range, the vehicles are able to maintain safe distances. Adapted with permission from [67] ©2021 IEEE.

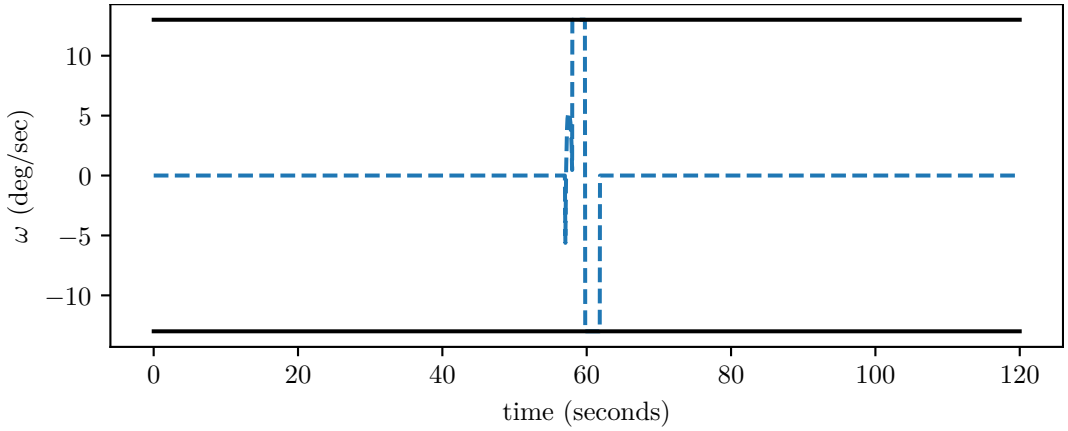
is shown in [69]. The vehicles are able to maintain safe distances throughout the simulation.

5.6 Conclusion

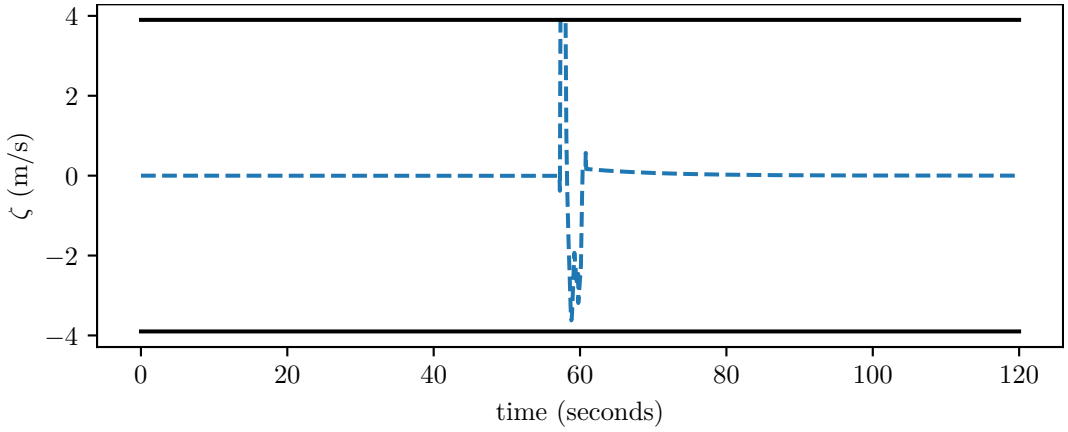
In this chapter we have found that safety guarantees are still possible even when there is limited range sensing. In order to arrive at this conclusion we construct a new barrier function from a previously existing one and ensure the new barrier function is constant outside the set of states that can be sensed. However, we note there are two downsides to this process. First, for any given barrier function it is not guaranteed that we can construct a new barrier function that can maintain safety when there is limited sensing. This is the case for $h_{straight}$ and shown more generally in Theorem 6. Second, because the state cannot be sensed outside of a given range, the resulting barrier function permits all control values that are within actuator constraints. The result is that the state may approach the safety boundary more quickly than if a longer range sensor were available. We summarize this discussion with an addition to Table 5.1.



(a)

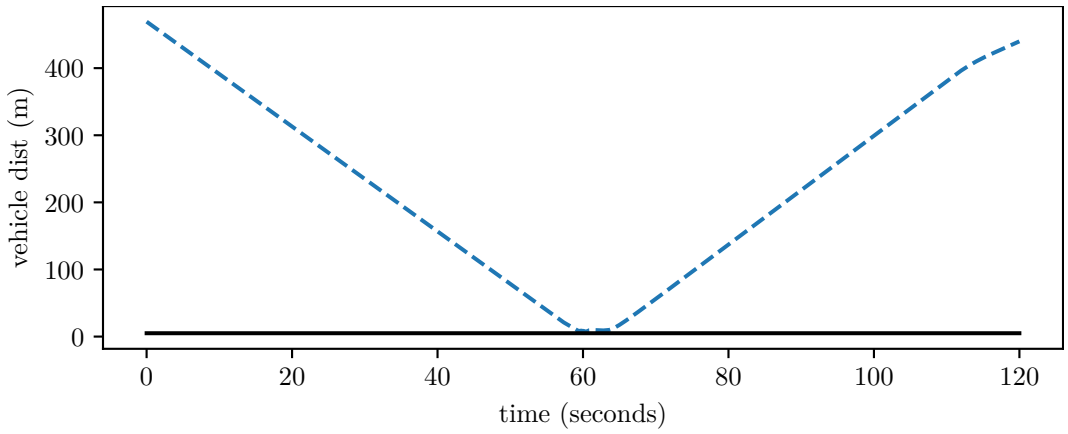


(b)

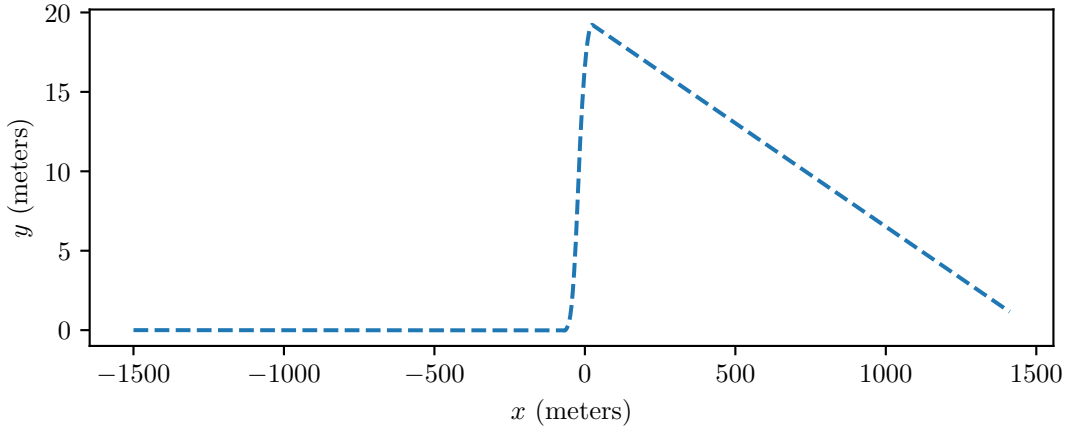


(c)

Figure 5.7: Control outputs for the scenario with 20 fixed-wing vehicles subject to limited sensing where h is constructed from γ_{turn}. Vehicle 1 velocity and turn rates are shown to be within the actuator limits in (a), (b), and (c) for velocity, turn rate, and altitude rate, respectively. Note that h constructed from $\gamma_{straight}$ is omitted because it has been shown that this h is not sensor compatible.

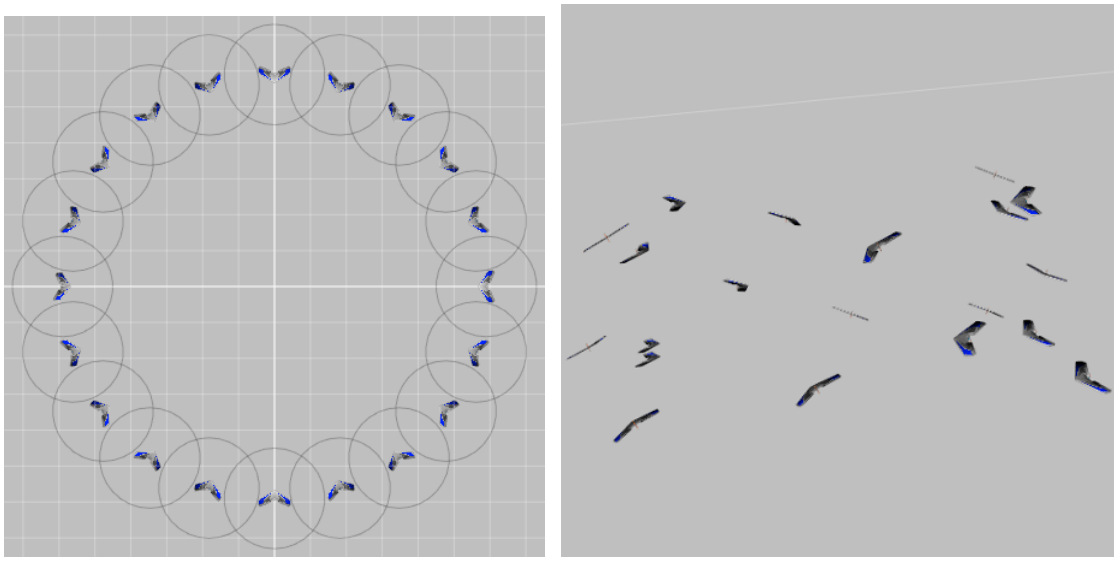


(a)



(b)

Figure 5.8: Outputs for the scenario with 20 fixed-wing vehicles subject to limited sensing where h is constructed from γ_{turn}. Vehicle 1 is plotted as a representative output since all 20 vehicles cannot be shown on the same plot. In (c), the minimum distance between any two vehicles is shown to be above D_s. (d) is the path taken by vehicle 1. Note that the behavior is significantly different when constructing h with γ_{turn} and $\gamma_{straight}$. Note that h constructed from $\gamma_{straight}$ is omitted because it has been shown that this h is not sensor compatible.



(a) (b)

Figure 5.9: (a) A screenshot of the initial conditions of 20 vehicles whose nominal controller will put them on a collision course through the origin. The circles around the vehicles are the sensing range so that the vehicles cannot sense each other at the beginning of the scenario. (b) A sideview of the aircraft as they maneuver around each other near the origin.

Table 5.1: A list of assumptions required to ensure system safety after adding the shared evading maneuver while removing the need to communicate and infinite range sensing.

Assumption	Effect of Removing Assumption
One Safety Constraint	still safe but additional assumption required (shared evading maneuver)
Unlimited Communication	still safe but overriding control value may deviate more significantly from nominal control value
Infinite Range Sensors	only some barrier functions can be used to guarantee safety and the state may approach the safety boundary more quickly than with unlimited sensing
Known Dynamics Model	see Chapter 6

CHAPTER 6

MODEL FREE SAFETY VIA IMPLICIT EVADING MANEUVERS

Chapters 3 through 5 removed assumptions of a single safety objective, unlimited communications, and unlimited sensing, respectively, made in Chapter 2. In particular, we found that safety could be guaranteed even when removing these assumptions. In this chapter we look at how to increase the performance goal. We find that we can increase performance by removing another assumption, namely that we have a known dynamics model. The result is to another method for constructing a barrier function that allows for a more performant system than a model based approach while still improving on the safety of the original system.

As shown in previous chapters, barrier functions can be used to maximize the performance of a system while satisfying a safety constraint. However, if the safety constraint arising from the barrier function is overly restrictive then performance can be needlessly diminished. For example, in an adaptive cruise control setting, safety designers can choose a minimum inter-vehicle distance that the vehicle must satisfy. Setting this minimum distance too high (e.g., hundreds of meters) will result in excessive inter-vehicle distances where speed setpoints are difficult to achieve. In other words, the performance goal (speed) is negatively impacted by an overly conservative constraint (inter-vehicle distances).

In this chapter we generalize this point by demonstrating that it extends beyond simple scalar settings like inter-vehicle distance. In particular, we consider the case of unmanned aerial vehicle (UAV) collision avoidance and demonstrate that even with a fixed minimum inter-vehicle safety distance, the observed behavior of the safety override resulting from a barrier function can be needlessly conservative. Specifically, we first consider the case where the barrier function safety constraint ensures each

vehicle can maintain a straight trajectory without a future collision. We show in this case that even when the vehicles are arbitrarily far apart that the barrier function indicates the vehicles are unsafe. This can result in significant performance implications on a waypoint-following UAV. To see this, consider another vehicle located far away and its effect on how well our own vehicle achieves its waypoints. The other vehicle could orient itself in a way that makes the barrier function imply an override is needed. This can make the system more unpredictable as non-local factors (e.g., vehicles far away) can have significant impact on the control choices. Further, it could even be exploited by malevolent actors who could conceivably choose to orient their own aircraft in a way that forces the waypoint-following aircraft to adjust and move in a way that a malevolent actor intends.

However, the concerns go further than that. We therefore consider a second case where a barrier function ensures the vehicles can employ a turning maneuver to maintain safe distances for all future time. However, in this case we construct a scenario where the nominal controller, a controller designed for a performance objective like waypoint following but that does not ensure safety, would result in inter-vehicle distances many times greater than the minimum safety threshold. Nevertheless, when an override from a barrier function is employed, the vehicles significantly alter their course in a way that causes them to barely exceed the minimum safety distance from each other. In other words, not only does the safety override needlessly alter the original system's control value, significantly reducing system performance, in doing so it also causes the vehicles to barely exceed the safety thresholds. Further, it can reduce trust in a system as observers see the safety override causing the system to fly needlessly close to the other vehicle.

Prior work has investigated how to relax the invasiveness of an override while ensuring that the system is always safe, often looking at how to construct a barrier function that specifically accounts for the nominal controller. For instance, in [50]

the authors introduce an optimization to maximize the set of safe states that are compatible with a region of attraction so that the safety constraint considers the performance objective. Similarly, an action policy and barrier function are learned simultaneously in [70]. Imitation learning has also been employed as in [71] where a barrier function is constructed from expert trajectories where the expert is able to consider both performance and safety factors. More broadly, barrier functions have also been used to not only enforce safety constraints but also guide exploration in [72] via off policy reinforcement learning. Similarly, in [73] the authors introduce a barrier function to constrain the policy update in reinforcement learning.

In this chapter, rather than simultaneously training both the nominal controller and safety override, we instead seek to maximize the set of safe controls available that could be applied to any nominal policy. This allows for a separation of concerns that simplifies the controller design process [42]– the nominal controller can be designed for performance while the safety override resulting from the barrier function overrides the nominal controller as little as possible while improving safety. In particular, we show that maximizing the set of safe states is not enough to ensure that a safety override is not restrictive. In other words, given a state that is safe for two different barrier functions, it may be that the available set of controls to keep the system safe is larger for a barrier function that has a smaller overall safe set.

We also construct a barrier function without requiring a dynamics model. A benefit of this is that it reduces the model mismatch that can occur when using simplified modeling assumptions (e.g., assuming linearity in the control input even though aircraft are subject to 6-DOF dynamics) that can lead to differences in simulated versus real-world performance. It has been more broadly discussed that model-free approaches can often outperform model-based systems [9] as they are less restricted in fitting to data. Thus, we propose model free barrier functions, which are learned from interactions with the environment, to reduce how much the system is overrid-

den. This chapter makes the following contributions. First, we motivate model free barrier functions with examples from fixed wing collision avoidance that demonstrates a model based approach induces unnecessary overrides. Second, we derive model free barrier functions. Third, we demonstrate the approach in simulation.

This chapter is organized as follows. Section 6.1 introduces the background for barrier functions. Section 6.2 derives model free barrier functions. Section 6.3 demonstrates the algorithm in simulation.

6.1 Background

In this section we introduce barrier functions in the context of discrete time [74] and compare model-free with model-based barrier functions. Thus we introduce a UAV dynamics model for two UAVs indexed by i ($i \in \{1, 2\}$) where the state for each UAV at time k is given by $x_{k,i} = \begin{bmatrix} p_{k,i,x} & p_{k,i,y} & \theta_{k,i} & p_{k,i,z} \end{bmatrix}^T$. The components $p_{k,i,x}$, $p_{k,i,y}$, and $p_{k,i,z}$ are the x, y, and z position of aircraft i and $\theta_{k,i}$ is the orientation. The discrete time dynamics are given by

$$x_{k+1,i} = \begin{bmatrix} p_{k,i,x} + v_{k,i} \cos\theta_{k,i} \Delta t \\ p_{k,i,y} + v_{k,i} \sin\theta_{k,i} \Delta t \\ \omega_{k,i} \Delta t \\ p_{k,i,z} + \zeta_{k,i} \Delta t \end{bmatrix}$$

where $v_{k,i}$ is the translational velocity, $\omega_{k,i}$ is the rotational velocity, $\zeta_{k,i}$ is the vertical velocity, and Δt is a integration step parameter. These velocities serve as control inputs and are subject to actuator limits v_{min} and v_{max} where $v_{min} > 0$, $|\omega_{k,i}| \leq \omega_{max}$, and $|\zeta| < \zeta_{max}$, respectively. Given two aircraft, the system with state $x_k = \begin{bmatrix} x_{k,1}^T & x_{k,2}^T \end{bmatrix}^T$ has dynamics of the form

$$x_{k+1} = f(x_k, u_k) \tag{6.1}$$

where $x_k \in \mathbb{R}^n$, $u_k \in U \subset \mathbb{R}^m$, and U is the set of available controls for the system. In [74] the authors consider dynamics of the form (6.1) to develop barrier functions for discrete time systems which we briefly summarize here. Let $h : \mathbb{R}^n \to \mathbb{R}$ be an output function of the state and define the safe set $\mathcal{C}$ as a superlevel set of h so that

$$\mathcal{C} = \{x_k \in \mathbb{R}^n \ : \ h(x_k) \geq 0\}. \tag{6.2}$$

We also let $\Delta h(x_k, u_k) = h(x_{k+1}) - h(x_k)$. When the output function h satisfies the following property it can be used to ensure that if the state starts in $\mathcal{C}$ then it will stay in $\mathcal{C}$ for all future time. The following definition is an adaptation from Definition 4 of [74] using terminology similar to that in [4].

Definition 4. A map $h : \mathbb{R}^n \to \mathbb{R}$ is a Discrete-Time Exponential Control Barrier Function (DT-ECBF) on a set $\mathcal{D}$ where $\mathcal{C} \subseteq \mathcal{D}$ if there exists a control input $u_k \in \mathbb{R}^m$ and λ such that

$$\Delta h(x, u) + \lambda h(x) \geq 0 \tag{6.3}$$

and $0 \leq \lambda \leq 1$ for all $x \in \mathcal{D}$.

We therefore define the admissible control space as

$$K(x_k) = \{u \in U \ : \ \Delta h(x_k, u_k) + \lambda h(x_k) \geq 0\}. \tag{6.4}$$

The following is an adaptation from Proposition 4 of [74] using the terminology of an admissible control space from [4].

Proposition 2. Given a set $\mathcal{C} \subset \mathbb{R}^n$ defined in (6.2) for an output function h, if h is a DT-ECBF on $\mathcal{D}$ then any controller $u : \mathbb{R}^n \to U$ such that $u(x) \in K(x)$ for all $x \in D$ will render the set $\mathcal{C}$ forward invariant.

Assuming the system has a controller designed to achieve a performance goal that does not necessarily ensure safety, barrier functions can be used to select a control

value u as close as possible to the original controller $\hat{u}$ while ensuring safety via the following optimization program:

$$\begin{aligned} u = \arg\min_{u \in U} \frac{1}{2} \|u - \hat{u}\|^2 \\ \text{s.t.} \quad u \in U \\ u \in K(x). \end{aligned} \tag{6.5}$$

As discussed in [74], equation (6.5) is a nonconvex program which can be difficult to solve in real-time. We resolve this runtime issue in this paper by assuming U is a discrete set small enough to calculate this optimization in real time. The final implementation is in a neural network where a single forward pass through a network can calculate many values for $K(x)$ in parallel as discussed for instance in [75].

6.2 Generating a Model Free Barrier Function via Evasive Maneuvers

6.2.1 Constructing Barrier Functions For Discrete Time Systems

Prior chapters have demonstrated a method for constructing a barrier function for continuous time systems. We therefore first adapt that method to discrete time. Let $\rho : \mathbb{R}^n \to \mathbb{R}$ be a safety function that must be nonnegative at all times for the system to be safe. Let $\gamma : \mathbb{R}^n \to U$ be an evasive maneuver that can keep the system safe. Note that γ is not necessarily the safety override but is instead used to construct a barrier function. To construct a candidate DT-ECBF we forward propagate the state using γ as the controller and calculate the worst case safety value using ρ for all future times. In other words, let

$$h(x_0) = \inf_{k \geq 0} \rho(\hat{x}_k) \tag{6.6}$$

be a candidate DT-ECBF where $\hat{x}_0 = x_0$ and $\hat{x}_k$ for $k > 0$ is the future state when using γ for all future time, namely

$$\hat{x}_{k+1} = f(\hat{x}_k, \gamma(\hat{x}_k)). \tag{6.7}$$

Using this formulation, we can show that h in (6.6) is a DT-ECBF. The theorem and proof are similar to Theorem 2 but applied to discrete time systems.

Theorem 7. Given a dynamical system (6.1) and a function h defined in (6.6) with a safety function ρ and an evasive maneuver γ, h is a DT-ECBF on the set $\mathcal{C}$.

Proof. Suppose $x \in \mathcal{C}$ so that $h(x) \geq 0$. Then

$$\Delta h(x, \gamma(x)) = \inf_{k \geq 1} \rho(\hat{x}_k) - \inf_{k \geq 0} \rho(\hat{x}_k).$$

The right hand side is nonnegative because it is the subtraction of the infimum of same function on different intervals where the first interval is a subset of the second interval. Then $\Delta h(x, \gamma(x)) \geq 0$. Recalling as well that $x \in \mathcal{C}$ means that $h(x) \geq 0$, this implies that $\Delta h(x, \gamma(x)) + \lambda h(x) \geq 0$. Then $\gamma(x) \in K(x)$ so h is a DT-ECBF. $\square$

Remark 16. Although in Definition 4 $\mathcal{D}$ can be a larger set than $\mathcal{C}$, Theorem 7 is only valid for $\mathcal{C} = \mathcal{D}$. See Theorem 2 for sufficient conditions for $\mathcal{C} \subset \mathcal{D}$ in the continuous time domain.

6.2.2 The Effect of The Evasive Maneuver on Safe Sets

While Theorem 7 shows that h defined in (6.6) is a DT-ECBF and can therefore be used to guarantee safety, it does not explicitly imply anything about the topology of the safe set $\mathcal{C}$ that is implied by h. In particular, for different choices of γ, the associated safe set can be drastically different.

Consider the two examples of a barrier function discussed in Section 2.3.2, namely h_{turn} and $h_{straight}$ as the h in (6.6) constructed from γ_{turn} and $\gamma_{straight}$, respectively. The reason these evasive maneuvers were considered previously is because they enable a closed form solution to (6.6) so that the barrier function constraint can be calculated in real-time. We consider some examples where the safe set implied by h_{turn} and $h_{straight}$ results in either an unnecessary invasive override or labeling states as unsafe that have ample room to avoid a collision. A graphical view of these two scenarios is in Figure 6.1. The actual path traversed by the vehicles for the scenario of Figure 6.1b is demonstrated in Figure 6.2.

Example 7. States Are Labelled Unsafe Where Collisions Can Be Avoided. Suppose h is parameterized by $\gamma_{straight}$ and consider an initial condition where the two vehicles are positioned at the same altitude with orientations pointing at each other. Then no matter how far apart the vehicles start, the calculation of (6.6) yields $h = -D_s$, implying that the initial conditions are not safe. This is because using $\gamma_{straight}$ as the evasive maneuver implies that the two vehicles will continue on a collision course until they collide. Clearly, as the vehicles are placed arbitrarily far apart, there is ample time to turn to avoid a collision. Nevertheless, according to $h_{straight}$, this configuration is outside of the safe set.

Example 8. An Unnecessary Invasive Override. While using h_{turn} resolves the issue raised in Example 7, there are other initial conditions that lead to an unnecessary override even when using γ_{turn}. For instance, suppose that the vehicles pass on the left of each other with a lateral separation of more than the safety distance but less than four turn radii. Then if the vehicles continue straight the vehicles will eventually approach an unsafe condition according to h_{turn} and the overriding safety controller will induce a large path correction so that each vehicle can pass on the others' left.

Examples 7 and 8 indicate $h_{straight}$ and h_{turn} may be restrictive. Another way of looking at this problem is to plot the set of unsafe states for a variety of configurations

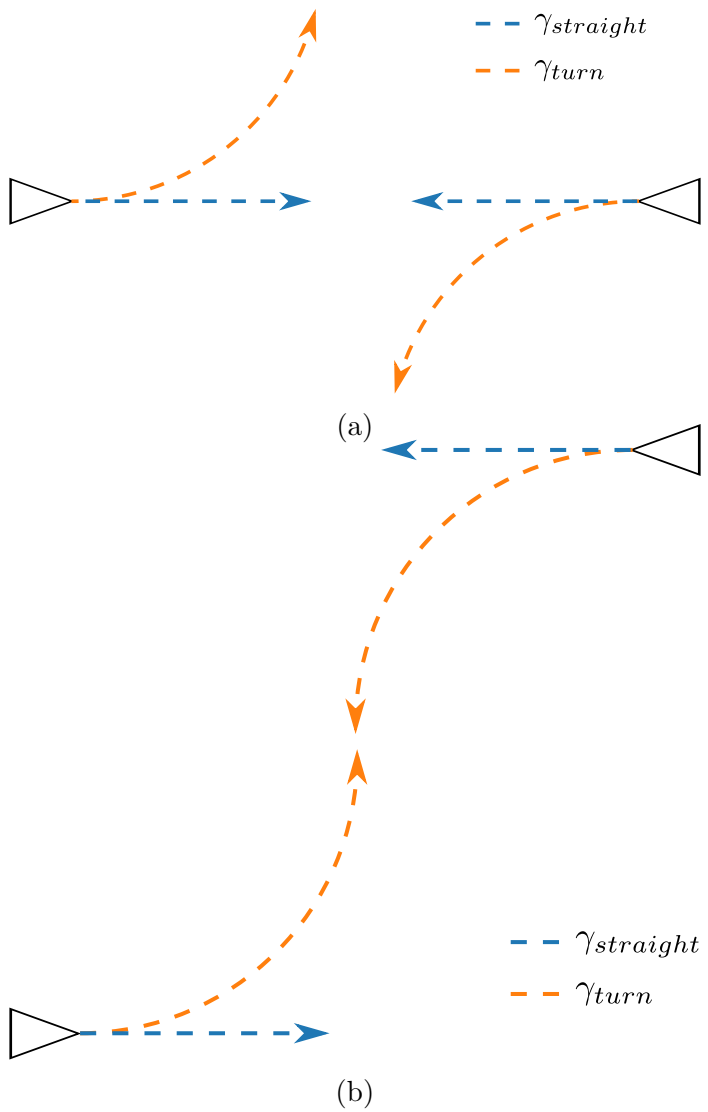


Figure 6.1: (a) When the two vehicles are pointing towards each other, the vehicles are not safe when using $h_{straight}$ no matter how far apart the vehicles start from each other. (b) When the vehicles pass to each others' left, the vehicles are not in the safe set when using h_{turn} because the evasive maneuver implies the vehicles would collide with each other. The practical takeaway in these two cases is that even though safety can be guaranteed, the particular states that are safe may be different. In either case though, collision avoidance can be achieved provided the vehicles start in a state such that $h(x) \geq 0$ for either $h_{straight}$ or h_{turn}.

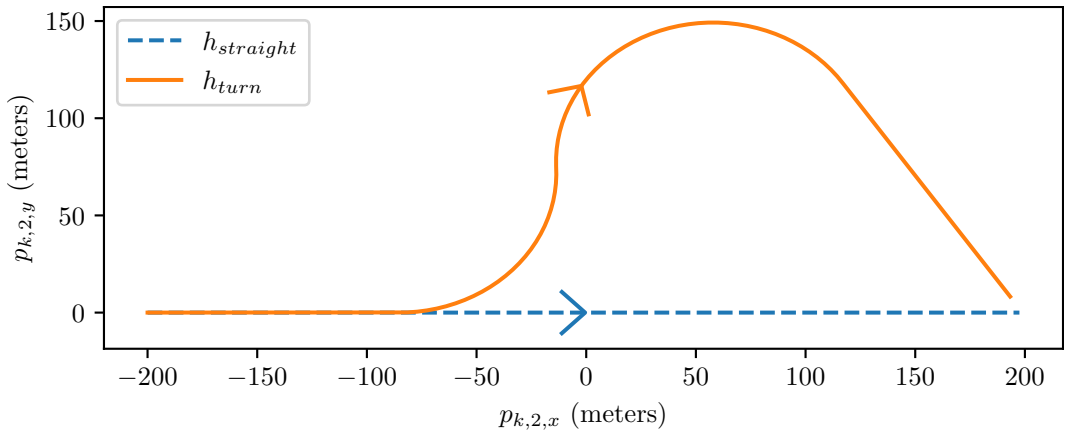


Figure 6.2: A plot of the trajectory of vehicle 2. When using $\gamma_{straight}$ to construct the barrier function, vehicle 2 does not deviate from the nominal control value. When using γ_{turn}, vehicle 2 deviates significantly. This is because the barrier function always makes sure that vehicle 2 can execute γ_{turn} to keep the vehicles safe even if there are better selections for an overriding controller (e.g., in this case, continuing straight). The trajectory for vehicle 1 is similar.

as is done in Figure 6.3. Figure 6.3 demonstrates that even when the vehicles are not pointing at each other, the vehicles can be spaced far apart and be in an unsafe state when using $\gamma_{straight}$ as the evasive maneuver. Further, Figure 6.3a even shows that the vehicles can be considered unsafe even when they have flown past each other when using h_{turn}. The point of these examples is that there are cases where a barrier function can induce overly restrictive safety overrides. We note that while these two examples demonstrate that a safety override using barrier functions may result in overly restrictive behavior this is not necessarily the case for all barrier functions. In particular, this chapter resolves these issues by finding a barrier function whose safe set is much larger than the safe set associated with either h_{turn} or $h_{straight}$ (see Fig. 6.5). It also shows using a maximum of barrier functions can increase the admissible control space.

6.2.3 An Initial Model Free Barrier Function

The issues in Figures 6.1 and 6.3 result because the evasive maneuvers used to calculate h are constant. More complicated evasive maneuvers might for instance involve

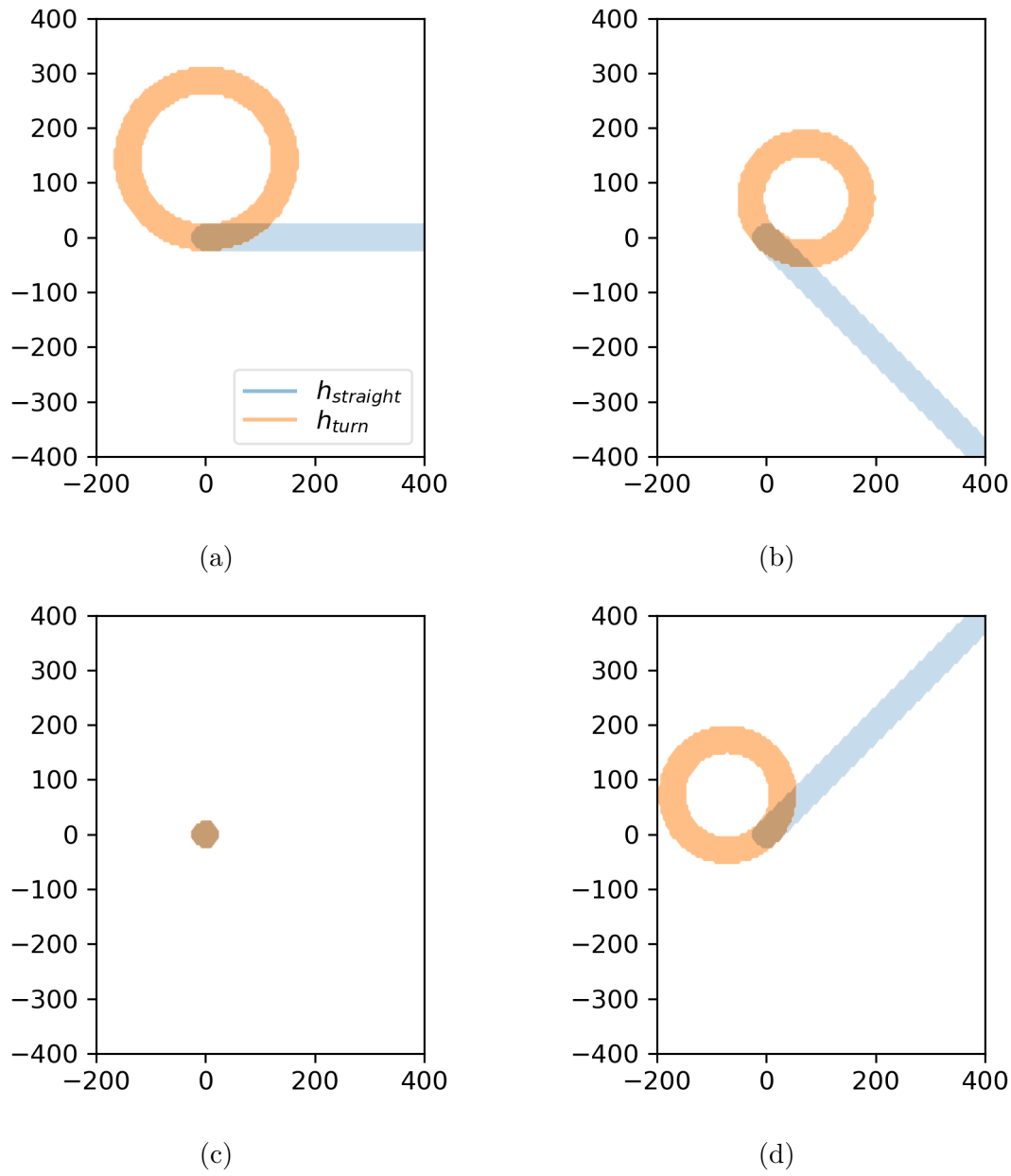


Figure 6.3: A plot of the unsafe set for four different configurations of the aircraft when using $h_{straight}$ (blue) or h_{turn} (orange). In all cases one of the vehicles is at the origin with orientation pointing along the positive x-axis. The plot shows when the set of states that are outside the safe set as the second vehicle changes x and y positions. In particular, it shows that some states are needlessly labelled unsafe. An example is in (a) where h_{turn} labels states as unsafe even when the vehicles have flown past each other. (a) The second vehicle is pointing left. (b) The second vehicle is pointing up. (c) The second vehicle is pointing right, (d) The second vehicle is pointing down.

turning right when the other vehicle is on the left and turning left when the other vehicle is on the right. More generally, we might expect that the evasive maneuver use the state to select an evasive maneuver which is not the case for $\gamma_{straight}$ and γ_{turn}. In this section we present a solution to this problem.

Another consideration not discussed in Section 6.2.2 for using Theorem 7 is that it requires that (6.6) be calculated over an infinite horizon. Nevertheless, as demonstrated in Section 2.3.2, h can be calculated in closed form for γ_{turn} and $\gamma_{straight}$. To do so, we assumed a particular model and chose an evading maneuver to enable closed form calculations. However, as shown in Section 6.2.2, this may result in safe sets that exclude many seemingly safe states.

More generally, it may be difficult to generate a barrier function if the system dynamics are complicated. Nevertheless, as demonstrated in Theorem 7, the evasive maneuver can be any function that maps from the state to the action space.

Because it may be difficult to calculate h in (6.6) in closed form for an arbitrary γ, we propose taking a data driven approach. To do so, we can start the state at some $x_0 \in \mathbb{R}^n$ and apply some evasive maneuver γ that we specify.[1] Because the nominal controller is available, we could for instance let $\gamma = \hat{u}$ and create a sequence $\{x_k\}_{k=0}^T$ where T is some horizon over which safety is evaluated. In the case of UAV collision avoidance, T may represent battery life of the vehicles where collisions will obviously not occur beyond time T.

Note that the sequence $\{x_k\}_{k=0}^T$ is the enumeration of the minimum on the right hand side of (6.6). Thus, given a starting state x_0, we can calculate $\rho_{min} = \min_{k\geq 0} \rho(x_k)$ to calculate a sample $h(x_0)$. Suppose this process is repeated N times to form a dataset $D = \{(x_0^j, \rho_{min}^j)\}_{j=1}^N$. Then we can fit a function $\hat{h}$ to approximate the mapping (6.6) with the dataset D. In the perfect case where there is no error in $\hat{h}$ we are

[1]Note that x_0 is sampled from $\mathbb{R}^n$ rather than $\mathcal{D}$ so that during the data-generation phase, the vehicles may start in an unsafe condition. This is to provide more data in fitting the learned $\hat{h}$. If the data did not include unsafe states then the data would have a bias towards predicting that states are safe.

left with a function that directly calculates (6.6) without having to do the integration in (6.6) because the integration is implicit in the fitting of the data.

However, when fitting $\hat{h}$ there will be errors. Errors where the learned $\hat{h}$ is less than the true h leads to more conservative behavior by considering states to be unsafe that are actually safe. On the other hand, when the learned $\hat{h}$ over predicts relative to the data, it can imply the state is safe when it is not. A conservative approach is to therefore bias the learned $\hat{h}$ downward to reflect uncertainty. This can be done by biasing the loss function as was done in [76] or alternatively with a Bayesian approach (e.g., Gaussian Processes [77] were used for barrier functions in [78]. Bayesian neural networks [79, 80] can also output an uncertainty) by subtracting a desired number of standard deviations learned model output. We note though that while this method reduces the chances that the fitted $\hat{h}$ will over predict the true h, because it cannot be guaranteed this type of error does not occur, the strict safety guarantee arising from of Theorem 7 is lost.

6.2.4 Iteratively Expanding the Admissible Control Space

Consider the output of $\hat{h}$ when applied to fixed wing collision avoidance with a waypoint finding nominal controller. Position two vehicles arbitrarily far apart with waypoints located at the starting position of the other vehicle, and orientations pointing at their respective waypoint. This configuration will be unsafe for $\hat{h}$ for the same reason as described in Example 7. We now show how to improve on this initial estimated $\hat{h}$ with an iterative algorithm.

We therefore examine the case where a barrier function h is available and seek to generate a new barrier function h^1 with a larger safe set than h. Given a state $x_0 \in \mathbb{R}^n$ and a nominal control value $\hat{u}$, let $\gamma^1 : \mathbb{R}^n \to U$ be the result of the optimization[2] in equation (6.5). We note that the function γ^1 can be used as an evasive maneuver

[2]Note that because x_0 is sampled from $\mathbb{R}^n$ rather than $\mathcal{D}$ it is not guaranteed that the optimization program has a solution when $x \notin \mathcal{D}$. This can be resolved for instance by adding a slack variable.

since it is a function that maps to the action space as required by Theorem 7. In other words, we can form a new barrier function h^1 via (6.6) such that

$$h^1(x_0) = \min_{k \geq 0} \rho(\hat{x}_k) \tag{6.8}$$

where

$$\hat{x}_{k+1} = f(\hat{x}_k, \gamma^1(\hat{x}_k)). \tag{6.9}$$

We denote the safe set of h^1 as $\mathcal{C}^1$.

Theorem 8. Given a dynamical system (6.1) let h be defined in (6.6) with safety function ρ and evasive maneuver γ. Let h^1 be defined in (6.8) with safety function ρ and evasive maneuver γ^1 defined as the output of (6.5). Then $\mathcal{C} \subseteq \mathcal{C}^1$.

Proof. Let $x_0 \in \mathcal{C}$. From Proposition 7, because γ^1 maps to values in $K(x)$ for all $x \in \mathcal{D}$, $\rho(\hat{x}_k) \geq 0$ for $k \geq 0$ where $\hat{x}_k$ is defined in (6.9). Then $h^1(x_0) \geq 0$. Then $x_0 \in \mathcal{C}^1$. □

Theorem 8 says that by using γ^1 rather than γ as the evasive maneuver, the safe set does not get smaller. We now show a case where $\mathcal{C}$ is a strict subset of $\mathcal{C}^1$.

Example 9. Consider a discrete double integrator system

$$x_{k+1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ \Delta t \end{bmatrix} u, \tag{6.10}$$

where $x_{k,1}$ is the position and $x_{k,2}$ is the velocity. Let $\rho(x_k) = x_{k,1}$ so that the system is point wise safe when the position is nonnegative. Let $\gamma(x) = 1$ and $\Delta t = 0.1$. Let $x_0 = \begin{bmatrix} 0.5 & -1 \end{bmatrix}^T$. Then $h(x_0) = -0.05$ so $x \notin \mathcal{C}$. In the case where $\hat{u} = 2 \in U$, the result of (6.5) is $\gamma^1(x) = 2$. Then using γ^1 to construct h^1 via (6.8), $h^1(x_0) = 0.2$ so $x_0 \in \mathcal{C}^1$.

The point of Example 9 is that γ^1 can in some cases do a better job at avoiding unsafe conditions and as a result the safety set is enlarged. However, as discussed in Section 6.2.3, there is a drawback to using γ^1. In particular, to apply Theorem 8, one then needs to forward propagate the dynamics (6.9) for all future time where the controller at every future timestep is the result of a nonconvex program (6.5) and return the minimum $\rho(x_k)$ for the resulting sequence $\{\hat{x}_k\}_{k=0}^T$. For online safety overrides, this is not computationally feasible. Thus, we pursue the data driven approach discussed in Section 6.2.3. The algorithm is described in Algorithm 1.

Algorithm 1: Initial algorithm for learning a model free barrier function.

```
input : h (barrier function), N (number of samples), û (nominal controller),
        T (safety horizon)
output: ĥ¹
Function ExpandSafeSet(h, ρ, N, T):
    D = {};
    for m ← 1 to N do
        select a random x₀;
        x ← x₀;
        ρ_min ← ρ(x);
        for j ← 1 to T do
            γ¹ ← from equation (6.5) using x, h, and û;
            x ← f(x, γ¹);
            ρ_min ← min(ρ_min, ρ(x));
        end
        append {x₀, ρ_min} to D;
    end
    ĥ¹ ←fit to D;
    return ĥ¹;
```

Given Theorem 8 and Example 9 and that there are no errors in fitting $\hat{h}^1$ to to h^1, we expect that $\mathcal{C}^1$ will be a superset of $\mathcal{C}$. However, notice that we can continue this process to form γ^2 with the property that $\gamma^2(x) \in K^1(x)$ for all $x \in \mathcal{C}^1$ where

$$K^1(x) = \{u \in U \ : \ \Delta\hat{h}^1(x) + \gamma\hat{h}^1(x) \geq 0\}. \tag{6.11}$$

This approach is summarized in Algorithm 2. In general, for a barrier function h^i we denote the admissible control space by K^i and the safe set by $\mathcal{C}^i$.

Algorithm 2: Algorithm for Iteratively Expanding the Safe Set

input : h (barrier function), N (number of samples), $\hat{u}$ (nominal controller), T (safety horizon), L (number of expansions)
output: $\hat{h}^L$
$\hat{h}^0 \leftarrow h$;
for $i \leftarrow 1$ to L do
  $\hat{h}^i \leftarrow$ExpandSafeSet($\hat{h}^{i-1}$, ρ, N, $\hat{u}$, T);
end

However, Algorithm 2 ignores a subtle issue, namely that it is not necessarily the case that $K^j(x) \subseteq K^i(x)$ for any $i > j$ even though Theorem 8 shows that $\mathcal{C}^j \subseteq \mathcal{C}^i$. This is demonstrated in the following example.

Example 10. Consider again the double integrator system in Example 9. Let $x_0 = \begin{bmatrix} 2 & -1 \end{bmatrix}$, $\gamma(x) = 0.5$, and $\lambda = 0.9$. Then a numerical calculation shows that $K(x_0) = \{u : u \geq -3.77\}$. Let

$$\hat{u}(x) = \begin{cases} 1 & x_{0,0} = 2 \text{ and } x_{0,1} = -1 \\ 0.5 & \text{otherwise} \end{cases}.$$

Then $K^1(x_0) = \{u : u \geq -3.67\}$. In other words, even though Theorem 8 shows that $\mathcal{C} \subseteq \mathcal{C}^1$, it is not the case that $K(x) \subseteq K^1(x)$.

Example 10 shows that even though the safe set is enlarged when using Algorithm 2, the set of controls available to keep the system safe may be reduced. This means that in some places of the safe set there may be a more aggressive safety override when using h^1 rather than h. For this reason we would like to use the maximum of the barrier functions h^j for $j \leq i$ in Algorithm 2. We note that the use of maximums for boolean composition of barrier functions for continuous time systems was previously analyzed in [55].

Theorem 9. Given a dynamical system (6.1) and DT-ECBFs h^1 and h^2, the function h^3 defined by $h^3(x) = \max(h^1(x), h^2(x))$ is a DT-ECBF on the set $\mathcal{C}^1 \cup \mathcal{C}^2$. Further, $K^1(x) \subseteq K^3(x)$ on the set $\mathcal{C}^1$ and $K^2(x) \subseteq K^3(x)$ on the set $\mathcal{C}^2$.

Proof. We first prove that h^3 is a DT-ECBF on $\mathcal{C}^1 \cup \mathcal{C}^2$. Suppose $x \in \mathcal{C}^1 \cup \mathcal{C}^2$ and without loss of generality, assume $h^1(x) \geq h^2(x)$ so $h^3(x) = h^1(x)$. Suppose $u \in U$ satisfies $\Delta h^1(x,u) + \lambda h^1(x) \geq 0$ and let $x_1 = f(x,u)$. Such a u exists because h^1 is a DT-ECBF. Then

$$\begin{aligned}
&\Delta h^3(x,u) + \lambda h^3(x) \\
&\quad = [\max(h^1(x_1), h^2(x_1)) - \max(h^1(x), h^2(x))] + \\
&\qquad \lambda \max(h^1(x), h^2(x)) \\
&\quad = \max(h^1(x_1), h^2(x_1)) - h^1(x) + \lambda h^1(x). \qquad (6.12)
\end{aligned}$$

Case 1: If $h^1(x_1) \geq h^2(x_1)$ then (6.12) becomes

$$\Delta h^3(x,u) + \lambda h^3(x) = \Delta h^1(x,u) + \lambda h^1(x) \geq 0.$$

Case 2: If $h^1(x_1) < h^2(x_1)$ then (6.12) becomes

$$\begin{aligned}
\Delta h^3(x,u) + \lambda h^3(x) &= h^2(x_1) - h^1(x) + \lambda h^1(x) \\
&\geq h^1(x_1) - h^1(x) + \lambda h^1(x) \\
&= \Delta h^1(x,u) + \lambda h^1(x) \geq 0.
\end{aligned}$$

Then h^3 is a DT-ECBF.

Note that this also establishes that $K^1(x) \subseteq K^3(x)$ on the set $\mathcal{C}^1$ under the condition that $h^1(x) \geq h^2(x)$. We now consider the case where $x \in \mathcal{C}^1$ and $h^1(x) <$

$h^2(x)$ to show that $K^1(x) \subseteq K^3(x)$ on the set $\mathcal{C}^1$. When $h^1(x) < h^2(x)$ we have

$$\begin{aligned}
&\Delta h^3(x,u) + \lambda h^3(x) \\
&\quad = [\max(h^1(x_1), h^2(x_1))] - h^2(x) + \lambda h^2(x) \\
&\quad \geq [\max(h^1(x_1), h^2(x_1))] - h^1(x) + \lambda h^1(x).
\end{aligned}$$

Continuing the same logic as cases 1 and 2 above we conclude $K^1(x) \subseteq K^3(x)$ on the set $\mathcal{C}^1$ under the condition that $h^1(x) < h^2(x)$. Then $K^1(x) \subseteq K^2(x)$ on $\mathcal{C}^1$. The case of $K^2(x) \subseteq K^3(x)$ on $\mathcal{C}^2$ is proven the same way. □

Remark 17. The optimization (6.5) is non-convex so finding an online solution may be computationally intensive. A direct solution to this is to assume U is a small finite set and allow the calculation to be done in a single forward pass of a neural network. However, an alternative occurs when h^1 is defined via (6.6) for some γ and $x \in \mathcal{C}^1$. Theorem 7 demonstrates that γ is always a feasible solution of (6.5) provided $h^1(x) \geq 0$ (and similarly for an evasive maneuver used to construct h^2 for $x \in \mathcal{C}^2$). Because $K^1(x) \subseteq K^3(x)$ for all $x \in \mathcal{C}^1$, this means that γ is a feasible solution for (6.5) when using h^3 and $x \in \mathcal{C}^1$.

Theorem 9 provides the justification for adjusting Algorithm 2 to use a maximum so that the safety set is enlarged as well as the admissible control space. However, the direct application of Theorem 9 implies that L barrier functions must be maintained, which implies memory growth and reduces online computation capability. For this reason, we elect to adjust the dataset accordingly in Algorithm 3. Note that the difference between the functions ExpandSafeSet and ExpandSafeSetWithMax occurs in line 18 in Algorithm 3 where the max is used.

Algorithm 3: Algorithm for Iteratively Expanding the Safe Set and the Admissible Control Space

input : h (barrier function), N (number of samples), $\hat{u}$ (nominal controller), T (safety horizon), L (number of expansions)
output: $\hat{h}^L$

$\hat{h}^0 \leftarrow h$;
for $i \leftarrow 1$ to L do
  $\hat{h}^i \leftarrow$ExpandSafeSetWithMax(h^{i-1}, ρ, N, T);
end
Function ExpandSafeSetWithMax(h, ρ, N, $\hat{u}$, T):
  $D = \{\}$;
  for $m \leftarrow 1$ to N do
    select a random x_0;
    $x \leftarrow x_0$;
    $\rho_{min} \leftarrow \rho(x)$;
    for $j \leftarrow 1$ to T do
      $\gamma^1 \leftarrow$ from equation (6.5) using x, h, and $\hat{u}$;
      ;
      $x \leftarrow f(x, \gamma^1)$;
      $\rho_{min} \leftarrow \min(\rho_{min}, \rho(x))$;
    end
  end
  append $\{x_0, \max(h(x_0), \rho_{min})\}$ to D;
  $\hat{h}^1 \leftarrow$fit to D;
  return $\hat{h}^1$;

6.2.5 Model Free Barrier Functions

While h^L in Algorithm 3 may appear to be model-free, a model is still required to select $\gamma^1(x)$ in line 12 because $K(x)$ requires a calculation of $\Delta h(x)$ which requires a model for the dynamics. Thus, to make the final result of Algorithm 3 model-free we must also create a learned function $\Delta\hat{h}$. To do so, we simply record the minimum $\rho(x)$ occurring after $j = 1$ in Algorithm 3.

6.3 SIMULATION EXPERIMENTS

In this section we validate the approach of Algorithm 3. We let $v_1 = v_2 = 15$ meters/second in equation (2.15) and restrict the action space of both agents to a selection of $[-12, 0, 12]$ degrees per second for ω while holding velocity fixed at 15 meters per second and altitude rate at 0. The initial state for each vehicle is between $\begin{bmatrix} -200 & -200 & -\pi & 0 \end{bmatrix}^T$ and $\begin{bmatrix} 200 & 200 & \pi & 0 \end{bmatrix}^T$. We also let $\rho(x) = \max(50, \sqrt{d_{1,2}(x)} - D_s)$. We use a ρ that is clipped at 50 rather than just $\sqrt{d_{1,2}(x)} - D_s$ to simplify data normalization so that the target values are not unbounded and note that this clipping does not change $\mathcal{C}$. Hyperparameters are listed in Table 6.1 and training statistics are plotted in Figure 6.4.

Similar to Figure 6.3, we plot in Figure 6.5 the safe set for the mean value of the model free barrier function as well as when three standard deviations are subtracted. When subtracting three standard deviations from the mean output of the model-free barrier function, the unsafe set is enlarged. We also plot how the unsafe set changes as the ExpandWithMax in Figure 6.6 to demonstrate that the safe set is enlarged as the algorithm proceeds.

We list the percentage of collisions at each training iteration in Table 6.2. Given the model-free barrier function at the given iteration we show what percentage of episodes the vehicles come within 25 meters of each other when the barrier function

Table 6.1: Hyperparameters when fitting a model free barrier function

Hyperparameter	Value
Learning Rate	1e-4
Batch Size	50000
Epochs	10000
Dropout Percent	50%
Num Samples for Stdev Calculation	50
Number of Fully Connected Layers	4
Width Per Layer	1024

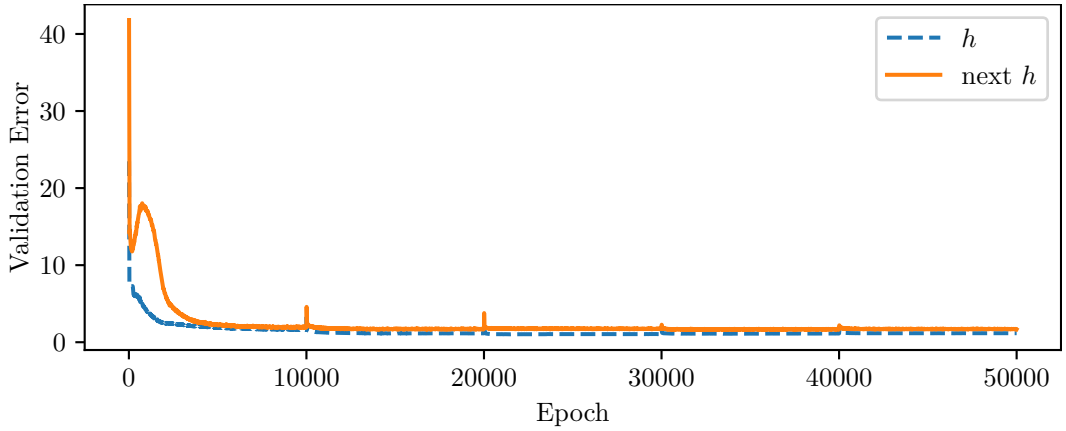
Table 6.2: The percentage of episodes where the vehicles collide from random initial conditions when the initial state is safe according to the barrier function versus the scenario where only the nominal controller is used. Notice that safety is significantly improved when using a model-based barrier function but that safety is nevertheless not guaranteed as there is noise in fitting to the data.

Iteration	Barrier	No Barrier
1	0.2	8.9
2	0.5	8.8
3	0.4	8.9
4	0.5	8.8
5	0.5	9.0

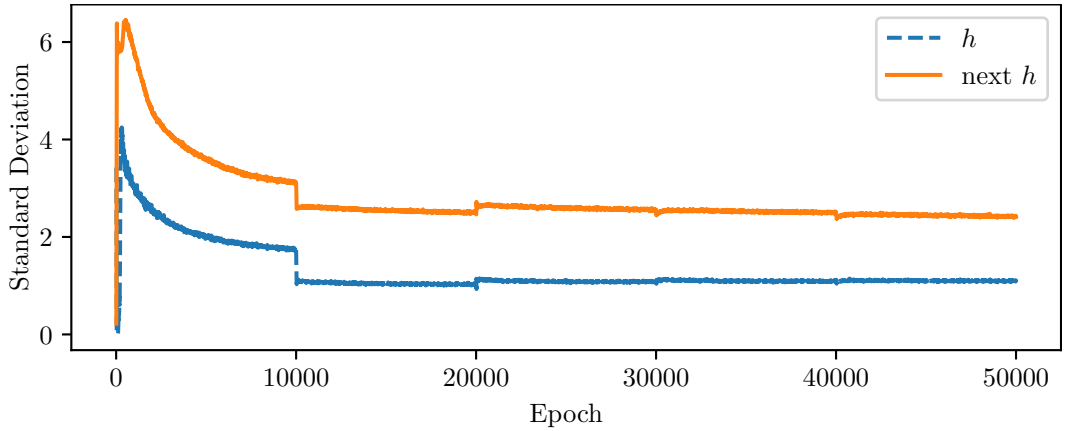
value of the first state is nonnegative. Notice that the number of collisions when using a model free barrier function is less than 10% of the number of collisions when using no barrier function. Additionally note that there are not zero collisions when using a model-free barrier function as there is noise in fitting to the data. Nevertheless, safety is significantly improved over using the nominal controller alone.

6.4 Conclusion

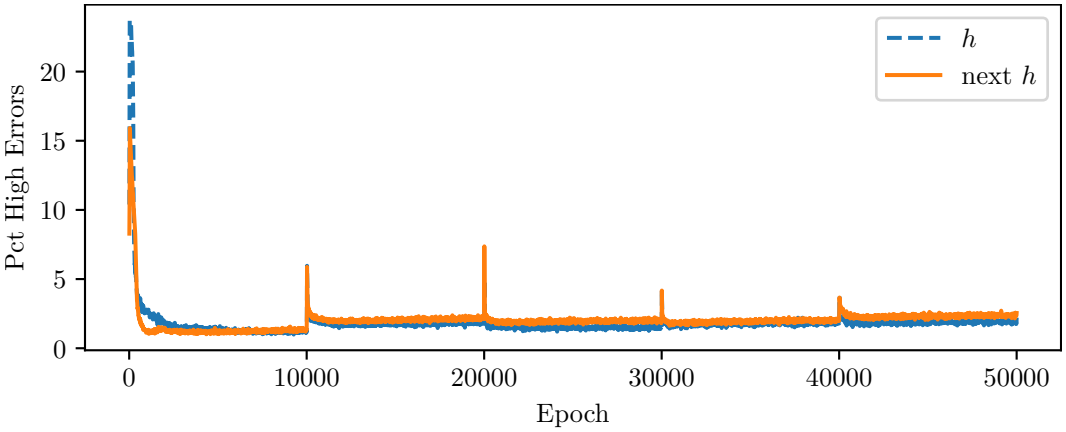
In this chapter we have discussed a few issues with model based barrier functions, namely that barrier functions may label safe states as unsafe (Example 7), barrier functions may cause unnecessary overrides that cause the state to get closer to the boundary of the safe set than without an override (Example 8), for complex systems it may be difficult to solve for a barrier function in closed form (h_{turn} and $h_{straight}$ exist due to closed form solutions but lead to large unsafe sets, see Fig. 6.5), and it can



(a)



(b)



(c)

Figure 6.4: Training data when fitting a model-free barrier function. (a) The validation error of the model-free network, (b) The standard deviation output by the network, and (c) How often the network outputs a value that is higher than the true value when subtracting 3 standard deviations from the mean of the network output.

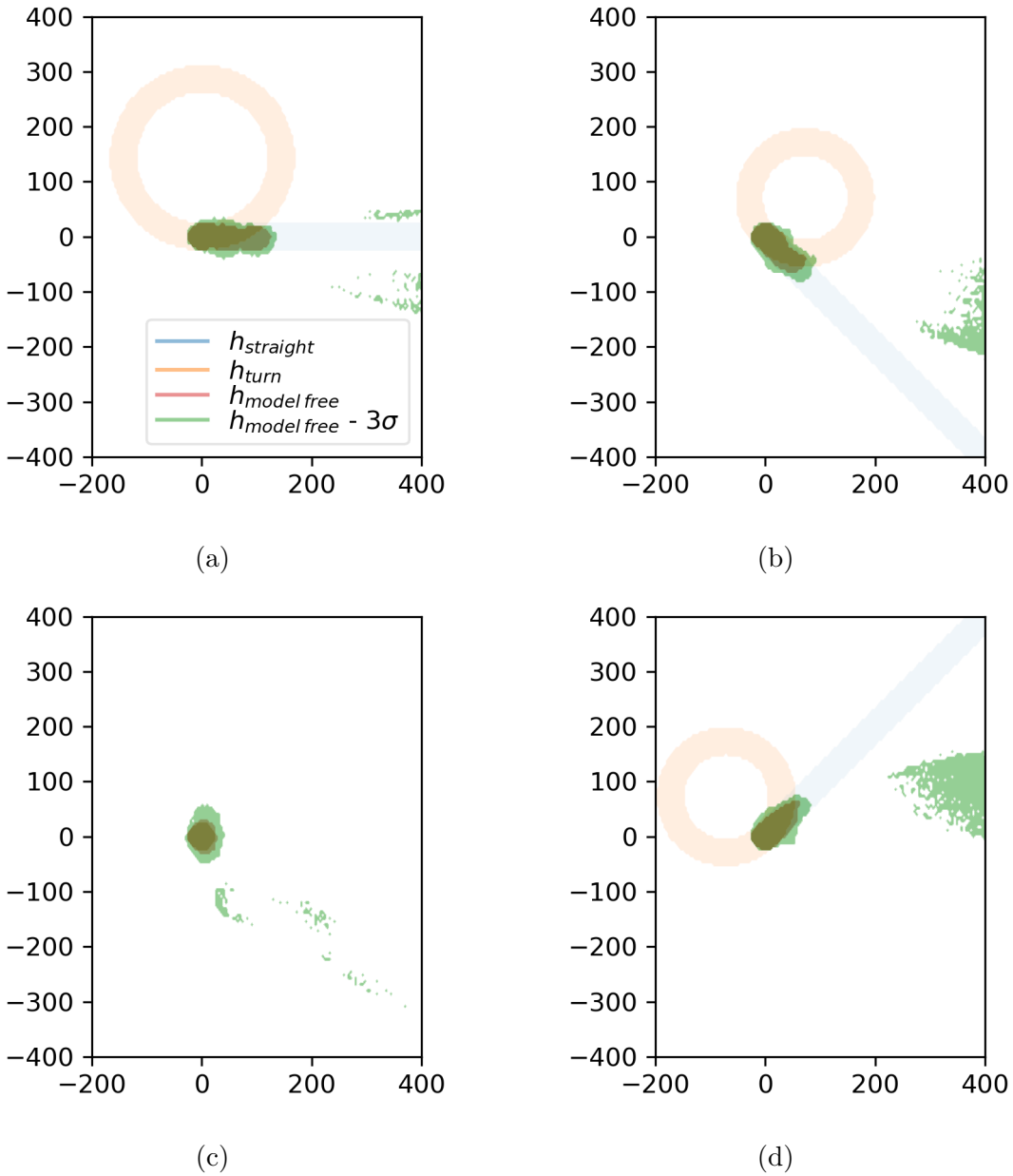


Figure 6.5: A plot of the unsafe set for the four different configurations of Fig. 6.3 for model based and a model free method. The model free method shows the mean output of a neural network as well as the case where we subtract three standard deviations from the mean. Note that the safe set for the mean output of the model free-barrier function is a subset of the model free unsafe sets. When subtracting 3 standard deviations from the model-free barrier function the unsafe set is larger than when using the mean. The data for fitting the model-free barrier function was sampled from positions (-200, -200) to (200, 200) so out-of-sample points are often labelled as unsafe due to the higher data uncertainty of those states. (a) The second vehicle is pointing left. (b) The second vehicle is pointing up. (c) The second vehicle is pointing right, (d) The second vehicle is pointing down.

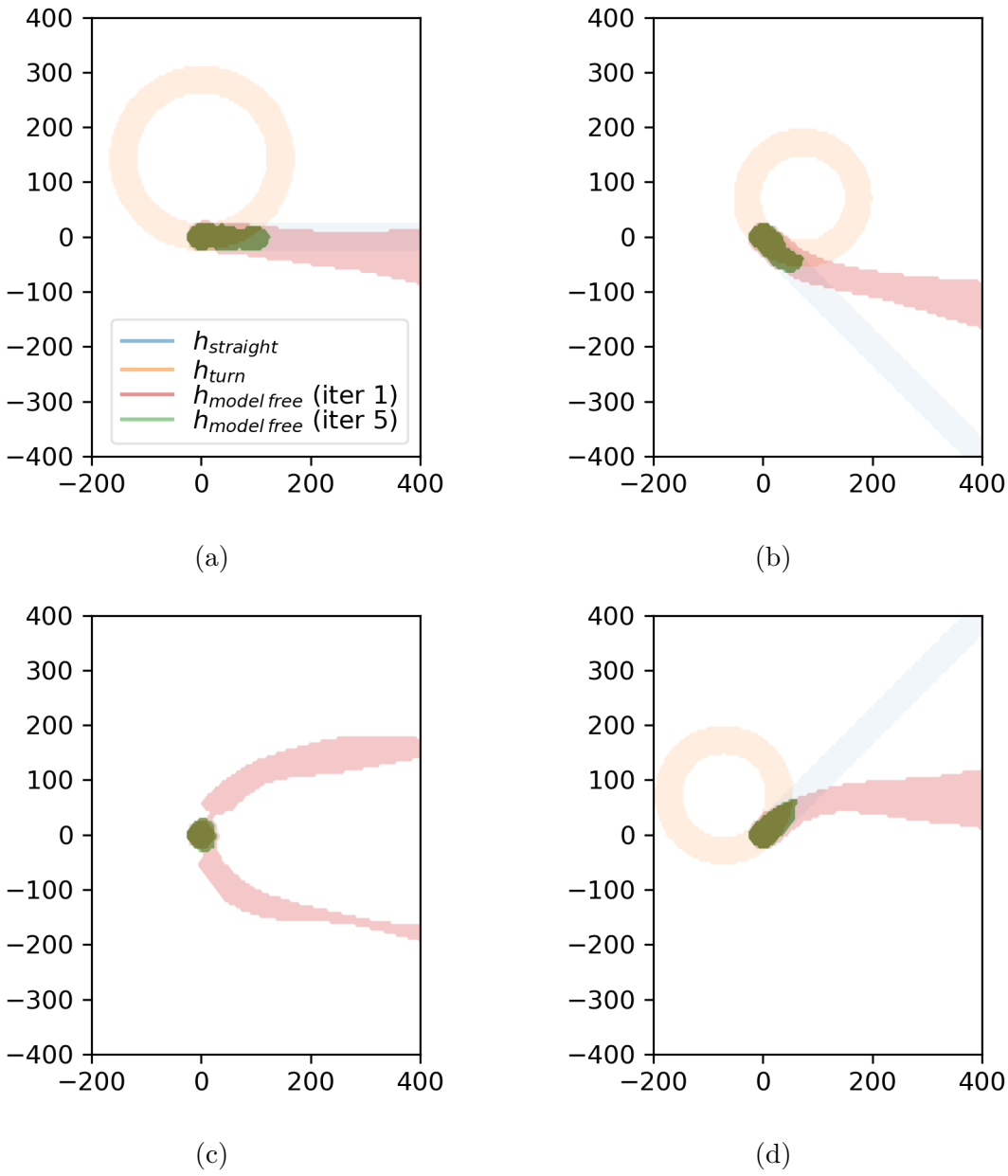


Figure 6.6: A plot of the unsafe set for the four different configurations of Fig. 6.3 for comparing early vs later iterations of the model free barrier functions. A plot demonstrating that the safe set grows as the algorithm proceeds. Note that on unsafe set of the barrier function at iteration 5 is a subset of the unsafe set at iteration 1, as predicted by Theorem 9. (a) The second vehicle is pointing left. (b) The second vehicle is pointing up. (c) The second vehicle is pointing right, (d) The second vehicle is pointing down.

Table 6.3: The effect assumptions have on safety when using barrier functions.

Assumption	Effect of Removing Assumption
One Safety Constraint	still safe but additional assumption required (shared evading maneuver)
Unlimited Communication	still safe but overriding control value may deviate more significantly from nominal control value
Infinite Range Sensors	only some barrier functions can be used to guarantee safety and the state may approach the safety boundary more quickly than with unlimited sensing
Known Dynamics Model	less invasive safety override but loss of safety guarantee

be numerically infeasible to solve for a barrier function when there is a long horizon (see equation (6.6)). To resolve these problems, we introduced model-free barrier functions which take a data-driven approach to developing a barrier function. The tradeoff is that because the barrier function cannot perfectly fit to the data, safety guarantees are lost but the benefit is that the safety set may be significantly enlarged (Fig. 6.5). Further, the optimization required to solve for a safe control input can be done in a single forward pass through a neural network. We demonstrated the efficacy of the approach in a fixed-wing aircraft collision avoidance scenario where, because of the model free barrier function, the safety of the system is significantly improved over using a nominal controller alone. We summarize this discussion in Table 6.3.

CHAPTER 7

CONCLUSION

In this thesis we have shown how to construct a minimally invasive framework to ensure safety while getting as much performance out of the system for arbitrarily many agents even with limited communication and sensing. In particular, the main contributions as as follows:

- Generalize a Method for Constructing a Barrier Function [63] - Given an evasive maneuver and a safety function that must be non-negative at all times for the system to be safe we construct a barrier function by forward propagating the dynamics using the evasive maneuver and calculate the smallest value along that trajectory to calculate a barrier function.

- Safety Composition [81] - We give sufficient conditions for multiple safety objectives to be satisfied for all future time.

- Limited Communication Safety [81] - We relax the assumption that each agent has knowledge of the other's control value and show that safety guarantees can still be made.

- Safety With Limited Range Sensing [67] - We give sufficient conditions to construct a barrier function in the context of limited range sensing that can be used to ensure the system stays safe for all times.

- Model Free Safety - We discuss limitations of model based barrier functions and introduce model free barrier functions to reduce the invasiveness of the safety override.

We demonstrate the contributions to barrier functions in a scenario of twenty fixed-wing aircraft whose nominal trajectories are designed to cause a collision but because of the proposed approach, are able to maintain safety and reach their goal location.

Appendices

APPENDIX A

AN ANALYSIS OF THE ROLE OF δ IN THE CONTINUOUS DIFFERENTIABILITY OF H_{TURN}

Note that (2.12) is not necessarily differentiable when $A_2 = 0$ since A_2 results from a square root performed in phasor addition. Thus, in this section, we consider how to ensure A_2 is continuously differentiable to ensure h in (2.12) is continuously differentiable. Consider (2.12) in phasor form

$$\begin{aligned} A_1 - D_s^2 + A_2 e^{j\Theta} &= A_1 - D_s^2 + \sigma A_3 e^{j(\theta_{1,0}-\pi/2)} + A_3 e^{j(\theta_{2,0}+\pi/2)} + \delta e^{j(\theta_{1,0}-\pi/2)} \\ &\quad + \sigma A_4 e^{j(\theta_{1,0}-\pi)} + A_4 e^{j\theta_{2,0}} + \delta e^{j(\theta_{1,0}-\pi)} \\ &= A_1 - D_s^2 + A_5 e^{j\Theta_5} + A_6 e^{j\Theta_6} \end{aligned} \tag{A.1}$$

where $A_3 = 2\Delta b_0 r$, $A_4 = 2\Delta c_0 r$, $A_5 e^{j\Theta_5} = \sigma A_3 e^{j(\theta_{1,0}-\pi/2)} + A_3 e^{j(\theta_{2,0}+\pi/2)} + \delta e^{j(\theta_{1,0}-\pi/2)}$, and $A_6 e^{j\Theta_6} = \sigma A_4 e^{j(\theta_{1,0}-\pi)} + A_4 e^{j\theta_{2,0}} + \delta e^{j(\theta_{1,0}-\pi)}$. Notice that $\Theta_5 - \Theta_6 = \pi/2$. In other words, A_2 is zero only when both A_5 and A_6 are zero. For $\delta = 0$, A_5 and A_6 are both zero on the set $Z_1 \subseteq \mathcal{D}$ where $\theta_{1,0} = \theta_{2,0}$ or $\theta_{1,0} = \theta_{2,0} + \pi$. Although Z_1 is a zero measure set, we note that for $\delta > 0$ that A_2 is zero on a set $Z_2 \subset Z_1$ where Z_2 is the restriction of Z_1 to a specific set of positions which we now specify.

Case 1. Vehicles Start in Opposite Directions. Suppose $\theta_{1,0} = \theta_{2,0} + \pi$. Then $A_5 = 0$ when $\delta = -(1+\sigma)A_3 = -2(1+\sigma)\Delta b_0 r$. Similarly, $A_6 = 0$ when $\delta = -(1+\sigma)A_4 = -2(1+\sigma)\Delta c_0 r$. Suppose δ is fixed. Then $A_2 = 0$ when $-\frac{\delta}{2(1+\sigma)r} = \Delta b_0 = p_{1,x_0} - p_{2,x_0} + r(1+\sigma)\sin\theta_{2,0}$ and $-\frac{\delta}{2(1+\sigma)r} = \Delta c_0 = p_{1,y_0} - p_{2,y_0} - r(1+\sigma)\cos\theta_{2,0}$.

Case 2. Vehicles Start in the Same Direction. Suppose $\theta_{1,0} = \theta_{2,0}$. Then $A_5 = 0$ when $\delta = (1-\sigma)A_3$. Similarly, $A_6 = 0$ when $\delta = (1-\sigma)A_4$. For $\sigma = 1$, let $\delta > 0$ to ensure A_5 and A_6 are not simultaneously 0. For $0 < \sigma < 1$, a similar analysis to the previous

case implies $A_2 = 0$ when when $-\frac{\delta}{2(1-\sigma)r} = \Delta b_0 = p_{1,x_0} + p_{2,x_0} - r(1-\sigma)\sin\theta_{2,0}$ and $-\frac{\delta}{2(1+\sigma)r} = \Delta c_0 = p_{1,y_0} - p_{2,y_0} - r(1+\sigma)\cos\theta_{2,0}$.

APPENDIX B

AN ANALYSIS OF THE CONTINUOUS DIFFERENTIABILITY OF $H_{STRAIGHT}$

From (2.16) we expand terms to get

$$h(x) = \inf_{\tau \in [0,\infty)} c(x) + b(x)\tau + a(x)\tau^2 \tag{B.1}$$

where $c(x) = \Delta x^2 + \Delta y^2 + \Delta z^2 - D_s^2$, $b(x) = 2(\Delta x \Delta C + \Delta y \Delta S)$, $a(x) = \Delta C^2 + \Delta S^2$, $\Delta x = p_{1,x_0} - p_{2,x_0}$, $\Delta y = p_{1,y_0} - p_{2,y_0}$, $\Delta z = p_{1,z_0} - p_{2,z_0}$, $\Delta C = v_1 \cos\theta_1 - v_2 \cos\theta_2$, $\Delta S = v_1 \sin\theta_1 - v_2 \sin\theta_2$. We also note that $a(x) > 0$ since

$$\begin{aligned} a(x) &= (v_1 \cos\theta_1 - v_2 \cos\theta_2)^2 + (v_1 \sin\theta_1 - v_2 \sin\theta_2)^2 \\ &= v_1^2 + v_2^2 - 2v_1 v_2 \cos(\theta_1 - \theta_2) \\ &= v_1^2 + v_2^2 - 2v_1 v_2 + 2v_1 v_2 - 2v_1 v_2 \cos(\theta_1 - \theta_2) \\ &= (v_1 - v_2)^2 + 2v_1 v_2 (1 - \cos(\theta_1 - \theta_2)) \\ &> 0 \end{aligned}$$

since $v_1 \neq v_2$ and v_1 and v_2 are positive. Then $\tau_{min}(x) = -b(x)/2a(x)$ is well defined. Then h has a minimum at $\tau_{nonneg,min} = \max(0, \tau_{min}(x))$.

For $\tau_{nonneg,min}(x) > 0$, h is continuously differentiable because c, b, τ_{min}, and a are continuously differentiable. Consider now when $\tau_{nonneg,min}(x) = 0$. We verify that $\frac{\partial h(x)}{\partial x} = \frac{\partial c(x)}{\partial x}$ for either the case of $\tau_{min} = 0$ or $\tau_{min} = -b(x)/2a(x)$. In the first case, $h(x) = c(x)$ and $\frac{\partial h(x)}{\partial x} = \frac{\partial c(x)}{\partial x}$. In the second case, $h(x) = c(x) + b(x)\tau_{min} + a(x)\tau_{min}^2$

and

$$\begin{aligned}\frac{\partial h(x)}{\partial x} &= \frac{\partial c(x)}{\partial x} + \frac{\partial b(x)}{\partial x}\tau_{min}(x) + b(x)\frac{\partial \tau_{min}(x)}{\partial x} + \frac{\partial a(x)}{\partial x}\tau_{min}(x) + 2a(x)\tau_{min}\frac{\partial \tau_{min}(x)}{\partial x} \\ &= \frac{\partial c(x)}{\partial x}\end{aligned}$$

because in this case $b(x)$ and $\tau_{min}(x)$ are 0.

REFERENCES

[1] M. Ben-Ari and F. Mondada, Elements of robotics. Springer Nature, 2017.

[2] A. G. Panaras, Aerodynamic principles of flight vehicles. American Institute of Aeronautics and Astronautics, 2012.

[3] Q. Nguyen and K. Sreenath, "Safety-critical control for dynamical bipedal walking with precise footstep placement," IFAC-PapersOnLine, vol. 48, no. 27, pp. 147–154, 2015.

[4] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," IEEE Transactions on Automatic Control, vol. 62, no. 8, pp. 3861–3876, 2017.

[5] E. Lalish, K. A. Morgansen, and T. Tsukamaki, "Decentralized reactive collision avoidance for multiple unicycle-type vehicles," in American Control Conference, 2008, IEEE, 2008, pp. 5055–5061.

[6] E. J. Rodriguez-Seda, "Decentralized trajectory tracking with collision avoidance control for teams of unmanned vehicles with constant speed," in American Control Conference (ACC), 2014, IEEE, 2014, pp. 1216–1223.

[7] P. Panyakeow and M. Mesbahi, "Decentralized deconfliction algorithms for unicycle uavs," in American Control Conference (ACC), 2010, IEEE, 2010, pp. 794–799.

[8] D. Gershgorn, The data that transformed ai research-and possibly the world, Quartz, Online; accessed 03/16/2021, https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world/, Jul. 2017.

[9] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 7559–7566.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., "Human-level control through deep reinforcement learning," nature, vol. 518, no. 7540, pp. 529–533, 2015.

[11] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, W. M. Czarnecki, A. Dudzik, A. Huang, P. Georgiev, R. Powell, et al., "Alphastar: Mastering the real-time strategy game starcraft ii," DeepMind blog, vol. 2, 2019.

[12] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al., "Dota 2 with large scale deep reinforcement learning," arXiv preprint arXiv:1912.06680, 2019.

[13] D. Ha and J. Schmidhuber, "World models," arXiv preprint arXiv:1803.10122, 2018.

[14] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," arXiv preprint arXiv:1912.01603, 2019.

[15] T. Weber, S. Racanière, D. P. Reichert, L. Buesing, A. Guez, D. J. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, et al., "Imagination-augmented agents for deep reinforcement learning," arXiv preprint arXiv:1707.06203, 2017.

[16] B. Thananjeyan, A. Balakrishna, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg, "Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks," IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 3612–3619, 2020.

[17] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[18] Z. Li, U. Kalabić, and T. Chu, "Safe reinforcement learning: Learning with supervision using a constraint-admissible set," in 2018 Annual American Control Conference (ACC), IEEE, 2018, pp. 6390–6395.

[19] W. Saunders, G. Sastry, A. Stuhlmueller, and O. Evans, "Trial without error: Towards safe reinforcement learning via human intervention," in Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 2067–2069.

[20] B. Eysenbach, S. Gu, J. Ibarz, and S. Levine, "Leave no trace: Learning to reset for safe and autonomous reinforcement learning," arXiv preprint arXiv:1711.06782, 2017.

[21] Z. Kenton, A. Filos, O. Evans, and Y. Gal, "Generalizing from a few environments in safety-critical reinforcement learning," arXiv preprint arXiv:1907.01475, 2019.

[22] T. Prevot, J. Rios, P. Kopardekar, J. E. Robinson III, M. Johnson, and J. Jung, "Uas traffic management (utm) concept of operations to safely enable low altitude flight operations," in 16th AIAA Aviation Technology, Integration, and Operations Conference, 2016, p. 3292.

[23] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Pérez, and J. Kuchar, "Collision avoidance for unmanned aircraft using markov decision processes," in AIAA guidance, navigation, and control conference, 2010, p. 8040.

[24] T. B. Wolf and M. J. Kochenderfer, "Aircraft collision avoidance using monte carlo real-time belief space search," Journal of Intelligent & Robotic Systems, vol. 64, no. 2, pp. 277–298, 2011.

[25] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," The International Journal of Robotics Research, vol. 17, no. 7, pp. 760–772, 1998.

[26] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," IEEE Robotics & Automation Magazine, vol. 4, no. 1, pp. 23–33, 1997.

[27] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in Robotics and Automation, 2007 IEEE International Conference on, IEEE, 2007, pp. 1986–1991.

[28] S. Mastellone, D. M. Stipanović, C. R. Graunke, K. A. Intlekofer, and M. W. Spong, "Formation control and collision avoidance for multi-agent non-holonomic systems: Theory and experiments," The International Journal of Robotics Research, vol. 27, no. 1, pp. 107–126, 2008.

[29] B. Di, R. Zhou, and H. Duan, "Potential field based receding horizon motion planning for centrality-aware multiple uav cooperative surveillance," Aerospace Science and Technology, vol. 46, pp. 386–397, 2015.

[30] M. Defoort, A. Kokosy, T. Floquet, W. Perruquetti, and J. Palos, "Motion planning for cooperative unicycle-type mobile robots with limited sensing ranges: A distributed receding horizon approach," Robotics and autonomous systems, vol. 57, no. 11, pp. 1094–1106, 2009.

[31] J. Shin and H. J. Kim, "Nonlinear model predictive formation flight," IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 39, no. 5, pp. 1116–1125, 2009.

[32] C. Tomlin, G. J. Pappas, and S. Sastry, "Conflict resolution for air traffic management: A study in multiagent hybrid systems," IEEE Transactions on automatic control, vol. 43, no. 4, pp. 509–521, 1998.

[33] C.-K. Lai, M. Lone, P. Thomas, J. Whidborne, and A. Cooke, "On-board trajectory generation for collision avoidance in unmanned aerial vehicles," in Aerospace Conference, 2011 IEEE, IEEE, 2011, pp. 1–14.

[34] Y. Lin and S. Saripalli, "Path planning using 3d dubins curve for unmanned aerial vehicles," in Unmanned Aircraft Systems (ICUAS), 2014 International Conference on, IEEE, 2014, pp. 296–304.

[35] ——, "Collision avoidance for uavs using reachable sets," in Unmanned Aircraft Systems (ICUAS), 2015 International Conference on, IEEE, 2015, pp. 226–235.

[36] I. Kolmanovsky, E. Garone, and S. Di Cairano, "Reference and command governors: A tutorial on their theory and automotive applications," in American Control Conference (ACC), 2014, IEEE, 2014, pp. 226–241.

[37] F. Tedesco, D. M. Raimondo, and A. Casavola, "Collision avoidance command governor for multi-vehicle unmanned systems," International Journal of Robust and Nonlinear Control, vol. 24, no. 16, pp. 2309–2330, 2014.

[38] D. Althoff, M. Althoff, and S. Scherer, "Online safety verification of trajectories for unmanned flight with offline computed robust invariant sets," in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 3470–3477.

[39] L. Pallottino, V. G. Scordio, A. Bicchi, and E. Frazzoli, "Decentralized cooperative policy for conflict resolution in multivehicle systems," IEEE Transactions on Robotics, vol. 23, no. 6, pp. 1170–1183, 2007.

[40] A. Krontiris and K. E. Bekris, "Using minimal communication to improve decentralized conflict resolution for non-holonomic vehicles," in Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, 2011, pp. 3235–3240.

[41] S. Prajna, "Barrier certificates for nonlinear model validation," Automatica, vol. 42, no. 1, pp. 117–126, 2006.

[42] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt, "Control barrier certificates for safe swarm behavior," IFAC-PapersOnLine, vol. 48, no. 27, pp. 68–73, 2015.

[43] L. Wang, A. D. Ames, and M. Egerstedt, "Safe certificate-based maneuvers for teams of quadrotors using differential flatness," in 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2017, pp. 3293–3298.

[44] S.-C. Hsu, X. Xu, and A. D. Ames, "Control barrier function based quadratic programs with application to bipedal robotic walking," in American Control Conference (ACC), 2015, IEEE, 2015, pp. 4542–4548.

[45] X. Xu, J. W. Grizzle, P. Tabuada, and A. D. Ames, "Correctness guarantees for the composition of lane keeping and adaptive cruise control," IEEE Transactions on Automation Science and Engineering, 2017.

[46] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Robustness of control barrier functions for safety critical control," IFAC-PapersOnLine, vol. 48, no. 27, pp. 54–61, 2015.

[47] X. Xu, T. Waters, D. Pickem, P. Glotfelter, M. Egerstedt, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Realizing simultaneous lane keeping and adaptive speed regulation on accessible mobile robot testbeds," in Control Technology and Applications (CCTA), 2017 IEEE Conference on, IEEE, 2017, pp. 1769–1775.

[48] L. Wang, A. D. Ames, and M. Egerstedt, "Multi-objective compositions for collision-free connectivity maintenance in teams of mobile robots," in Decision and Control (CDC), 2016 IEEE 55th Conference on, IEEE, 2016, pp. 2659–2664.

[49] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in HSCC, Springer, vol. 2993, 2004, pp. 477–492.

[50] L. Wang, D. Han, and M. Egerstedt, "Permissive barrier certificates for safe stabilization using sum-of-squares," arXiv preprint arXiv:1802.08917, 2018.

[51] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," Mathematical programming, vol. 96, no. 2, pp. 293–320, 2003.

[52] Q. Nguyen and K. Sreenath, "Exponential control barrier functions for enforcing high relative-degree safety-critical constraints," in American Control Conference (ACC), 2016, IEEE, 2016, pp. 322–328.

[53] X. Xu, "Constrained control of input–output linearizable systems using control sharing barrier functions," Automatica, vol. 87, pp. 195–201, 2018.

[54] T. Gurriet, M. Mote, A. D. Ames, and E. Feron, "An online approach to active set invariance," in 2018 IEEE Conference on Decision and Control (CDC), IEEE, 2018, pp. 3592–3599.

[55] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," IEEE control systems letters, vol. 1, no. 2, pp. 310–315, 2017.

[56] L. Wang, A. Ames, and M. Egerstedt, "Safety barrier certificates for heterogeneous multi-robot systems," in American Control Conference (ACC), 2016, IEEE, 2016, pp. 5213–5218.

[57] G. Wu and K. Sreenath, "Safety-critical control of a 3d quadrotor with range-limited sensing," in ASME 2016 Dynamic Systems and Control Conference, American Society of Mechanical Engineers, 2016, V001T05A006–V001T05A006.

[58] R. Konda, A. D. Ames, and S. Coogan, "Characterizing safety: Minimal control barrier functions from scalar comparison systems," IEEE Control Systems Letters, 2020.

[59] K. DeMarco, E. Squires, M. Day, and C. Pippin, "Simulating collaborative robots in a massive multi-agent game environment (SCRIMMAGE)," in Int. Symp. on Distributed Autonomous Robotic Systems, 2018.

[60] R. Olfati-Saber, "Near-identity diffeomorphisms and exponential/spl epsi/-tracking and/spl epsi/-stabilization of first-order nonholonomic se (2) vehicles," in American Control Conference, 2002. Proceedings of the 2002, IEEE, vol. 6, 2002, pp. 4690–4695.

[61] L. J. Clancy, Aerodynamics. Halsted Press, 1975.

[62] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," ArXiv e-prints, Nov. 2017. arXiv: 1711.08013 [math.OC].

[63] © 2018 IEEE, Reprinted, with permission, from Eric Squires, Pietro Pierpaoli, and Magnus egerstedt, "Constructive Barrier Certificates With Applications To Fixed-Wing Aircraft Collision Avoidance", Aug 2018.

[64] F. Blanchini and S. Miani, Set-theoretic methods in control. Springer, 2008.

[65] E. Squires, Composition of safety constraints for fixed-wing collision avoidance amidst limited communications, https://youtu.be/5y0l5taoJw4, Accessed: 2020-02-10, 2020.

[66] A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, and A. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," arXiv preprint arXiv:2010.09819, 2020.

[67] © 2021 IEEE, Reprinted, with permission, from Eric Squires, Rohit Konda, Pietro Pierpaoli, Samuel Coogan, and Magnus egerstedt, "Safety With Limited Range Sensing Constraints For Fixed Wing Aircraft", Jun 2021.

[68] K. K. Hassan, Nonlinear systems. Prentice Hall, Upper Saddle River, NJ 07458, 2002.

[69] E. Squires, Safety with limited range sensing constraints for fixed wing aircraft, https://youtu.be/SJp6zBZfB0A, Accessed: 2021-03-09, 2021.

[70] Z. Qin, K. Zhang, Y. Chen, J. Chen, and C. Fan, "Learning safe multi-agent control with decentralized neural barrier certificates," arXiv preprint arXiv:2101.05436, 2021.

[71] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in 2020 59th IEEE Conference on Decision and Control (CDC), IEEE, 2020, pp. 3717–3724.

[72] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 3387–3395.

[73] H. Ma, J. Chen, S. E. Li, Z. Lin, Y. Guan, Y. Ren, and S. Zheng, "Model-based constrained reinforcement learning using generalized control barrier function," arXiv preprint arXiv:2103.01556, 2021.

[74] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation.," in Robotics: Science and Systems, 2017.

[75] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," arXiv preprint arXiv:1705.08926, 2017.

[76] M. Srinivasan, A. Dabholkar, S. Coogan, and P. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," arXiv preprint arXiv:2003.04950, 2020.

[77] C. E. Rasmussen and C. K. Williams, Gaussian processes for machine learning. MIT press Cambridge, 2006, vol. 1.

[78] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 2460–2465.

[79] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," arXiv preprint arXiv:1505.05424, 2015.

[80] Y. Gal, "Uncertainty in deep learning," University of Cambridge, vol. 1, no. 3, 2016.

[81] E. Squires, P. Pierpaoli, R. Konda, S. Coogan, and M. Egerstedt, "Composition of multiple safety constraints with applications to decentralized fixed-wing collision avoidance," arXiv preprint, 2018.

VITA

Eric Squires is a Research Engineer in the Robotics and Autonomous Systems Division of the Aerospace, Transportation and Advanced Systems Lab at the Georgia Tech Research Institute (GTRI) where he leads research into machine learning algorithms. During that time he has focused on reinforcement learning and has developed novel algorithms for improving out of sample performance, trained optimal agents for aerial maneuvers, introduced novel solutions for multi-agent reinforcement learning, and shown how to incorporate safety into optimal systems. Additionally he has developed algorithms in multi-agent systems, specifically for fixed-wing aircraft, in both path planning and controls. Recent work in multi-agent control has shown how to use a barrier function to keep teams of fixed wing aircraft safe from collisions. Eric completed his Masters of Electrical and Computer Engineering at the Georgia Institute of Technology in 2015 and is currently a PhD candidate at Georgia Tech.

Prior to to joining GTRI, Mr. Squires worked at Ronald Blue & Co as a Senior Investment Analyst. During that time he focused on statistical analysis for economic data and software development. He was also a member of the Investment Policy Committee (IPC) and chaired the IPC Systemic Risk Subcommittee to identify low probability but high impact risks to portfolio performance.

As a result of his work, Eric has received numerous awards including the Ronald Blue & Co President's Award in 2011, the GTRI Collaboration Award in 2014, the Spot Award in 2015, and the Outstanding Technical Achievement Award in 2017. Eric is also a member of IEEE. He currently resides in Georgia and was born in California.