

**SAFE CONTROL OF PARTIALLY UNKNOWN SYSTEMS
LEVERAGING EFFICIENT REACHABILITY**

A Dissertation
Presented to
The Academic Faculty

By

Michael Enqi Cao

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

Georgia Institute of Technology

December 2025

© Michael Enqi Cao 2025

**SAFE CONTROL OF PARTIALLY UNKNOWN SYSTEMS
LEVERAGING EFFICIENT REACHABILITY**

Thesis committee:

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

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

Dr. Matthieu Bloch
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ye Zhao
Mechanical Engineering
Georgia Institute of Technology

Dr. Kyriakos Vamvoudakis
Aerospace Engineering
Georgia Institute of Technology

Date approved: August 14th, 2025

...[people] get on my nerves with the old “well, science doesn’t know everything.” Well, science *knows* it doesn’t know everything, otherwise, it’d stop! That doesn’t mean you can just fill in the blanks with whatever fairytale most appeals to you!

Dara O’Brian

For my parents, Fengli (Frank) and Zhixin (Tracy), who always encourage me to seek my own path, and my sister, Michelle, who fully embodies that spirit.

ACKNOWLEDGMENTS

First, I would like to thank the members of my thesis committee for their wisdom and assistance in preparing this work. I would like to thank my advisor, Dr. Samuel Coogan, for being an invaluable mentor throughout the Robotics PhD program, not only for his expertise in the field but also for his guidance along the path to becoming a capable researcher. I'd also like to thank Dr. Matthieu Bloch for being a wonderful collaborator, whose knowledge and enthusiasm were key, and thus, without whom several of these chapters would not exist. Thank you to Dr. Kyriakos Vamvoudakis, whose interest in the work ensured that the future holds exciting prospects, and Dr. Yorai Wardi, whose philosophical probing ensured that sight of the bigger picture is never lost. Last but certainly not least, thank you to Dr. Ye Zhao, whose mechanical expertise provided new perspectives from which to view and grow these ideas.

To the members of the FACTS Lab who came before: Max, Mohit, Cesar, Qinshuang, Chris, Matt, Gustav, and Saber; thank you for welcoming a fresh-faced grad student who had no idea what to expect, and making him feel both valued and confident that this journey would be worthwhile. To those whom I have worked alongside: Carmen, Soobum, Akash, Christian, James, Evanns, Luke, Jesse, and Brendan; it has been an honor and a privilege to see the future of the lab be shaped in front of my eyes, and I cannot wait to see what you accomplish in the future.

To my friends: Tony, Michael, Lemons, Ashwin, and far too many more to list here; thank you for being there, as a source of joy and relief, when it felt as though this journey would never end. Indeed, to anyone who has shown me support over the years, I truly could not have made it to this point without you all, and I cannot overstate how much it meant to me. To my family: Frank, Tracy, and Michelle, thank you for your encouragement and love, and your assurance that, even though home might physically be far away, it somehow always remains close by, regardless. I hope I did you all proud, and I hope to one day be able to return the favor.

TABLE OF CONTENTS

Acknowledgments	v
List of Tables	x
List of Figures	xi
List of Acronyms	xvi
Summary	xvii
Chapter 1: Safe Control of Partially Unknown Systems Leveraging Efficient Reachability: An Introduction	1
Chapter 2: Estimating High Probability Reachable Sets using Gaussian Processes . . .	4
2.1 Introduction	4
2.2 Notation	6
2.3 Problem Formulation	7
2.4 High Probability Reachable Sets	9
2.4.1 Reachable Sets	10
2.4.2 Forward Invariant Sets	13
2.4.3 Refinement of Reachable and Forward Invariant Sets	18
2.5 Gaussian Processes for High Probability Uncertainty Bounds	22

2.6	Case Studies	27
2.6.1	Case Study: Vehicle on an Icy Road	27
2.6.2	Example: Equilibrium Sequence Convergence	33
2.6.3	Case Study: Multirotor Exploring a Wind Field	34
Chapter 3: Safe Learning-based Predictive Control from Efficient Reachability		39
3.1	Introduction	39
3.2	Problem Setup	40
3.3	Safe Learning Algorithm	42
3.3.1	Mixed Monotonicity	42
3.3.2	Safe With High Probability MPC	43
3.4	Case Studies	46
3.4.1	Autonomous Vehicle on Icy Road	47
3.4.2	Planar Multirotor in Wind Field	50
3.4.3	Boat on a River	53
Chapter 4: An Optimistic Approach to Cost-Aware Predictive Control		56
4.1	Introduction	56
4.2	Related Work	58
4.3	Problem Setup	60
4.4	An Optimistic Control Algorithm	63
4.4.1	Theoretical Results from a Simplified Setting	66
4.5	Case Study Results	70

Chapter 5: Safe and Performant Control via Efficient Overapproximation of the Reachable Set Probability Distribution	74
5.1 Introduction	74
5.2 Problem Setup	76
5.3 Overapproximating the Probability Distribution of Reachable Sets	77
5.4 A Control Algorithm to Balance Performance and Safety	80
5.5 Case Studies	81
5.5.1 One-Shot Monte Carlo	83
5.5.2 Model Predictive Control	85
Chapter 6: Trajectory Tracking Runtime Assurance for Systems with Partially Unknown Dynamics	87
6.1 Introduction	87
6.2 Problem Setup	90
6.3 High Probability Forward Invariance	91
6.3.1 Mixed Monotonicity	91
6.3.2 Forward Invariant Tube	92
6.4 Aerial Vehicle Case Study	95
Chapter 7: Trajectory Tracking for Systems with Unknown Time-Varying Disturbances	99
7.1 Introduction	99
7.2 Problem Setup	99
7.3 Time-Varying Gaussian Process Bounds	100
7.4 Feedback Aware Forward Invariance	102
7.5 Case Studies	107

7.5.1	Numerical Example	107
7.5.2	Trajectory Tracking Runtime Assurance	108
7.5.3	Implicit Active Set Invariance Filtering	111
	References	113
	Vita	121

LIST OF TABLES

4.1	Statistical summary of the incurred cost of the Pessimistic strategy as well as the Optimistic strategies tested with different minimum required probabilities.	73
5.1	Results from 1000 runs in the Monte Carlo simulation, where the number of runs in which each strategy hit the goal and/or hit the obstacle are recorded. The Full strategy performs better than the Conservative strategy while maintaining the same safety level. While the Reckless strategy achieves the most goal hits, it is also the only strategy that hits the obstacle. We also note that the Full strategy, despite considering four levels of probability, takes less than four times the amount of computation time compared to the other strategies.	85
5.2	Average step computation time, number of timesteps to goal, and total acceleration applied from 10 runs in the MPC simulation. As shown, the Full strategy arrives at the goal in the fewest number of timesteps and is able to apply the largest amount of acceleration, while not being significantly more computationally expensive. . . .	86

LIST OF FIGURES

1.1	Systems with partially unknown dynamics, and strategies to enable safe operation.	1
2.1	The planar bicycle model has positions X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(\delta_f)$, front and rear distances from center of mass l_f and l_r . The inputs to the system are the desired acceleration u_1 and front steering angle $u_2 = \delta_f$	28
2.2	Known acceleration and steering angle inputs to the vehicle.	29
2.3	Visualization of the hyperrectangular overapproximation of forward reachable set using mixed monotonicity in the autonomous vehicle case study. The disturbance affecting the system is such that the true trajectory of the system notably differs from the undisturbed trajectory (left column, dotted lines). Initially, the system has low knowledge of the disturbance behavior, resulting in a conservative overapproximation of the true reachable set of the vehicle (top row). After obtaining several observations, the reachable set overapproximation is able to tightly approximate the true reachable set (bottom row).	32
2.4	Demonstration of Theorem 4 using embedding system (Equation 2.82) and disturbance bounds (Equation 2.79) and (Equation 2.80), $m = 2$. $[x^1, \bar{x}^1]$ is initialized to $[(-7, -7), (7, 7)]$, shown in red (outermost solid), and the subsequent steps in the sequence are plotted in black (dashed), with the final converged hyperrectangle shown in green (innermost solid). At $k = 2$, the sequence fulfills (Equation 2.26), from which we conclude that the remaining subsequence is ordered and converges from the second part of Theorem 4. The resulting hyperrectangle, $[(-1.72, -3.40), (1.72, 3.40)]$ is an equilibrium of (Equation 2.82) and is thus forward invariant for (Equation 2.78) per Theorem 4.	34

2.5	<p>Demonstration of Proposition 1 using embedding system (Equation 2.82) and disturbance bounds (Equation 2.79) and (Equation 2.80), $m = 1$. Note that $(\underline{\gamma}, \bar{\gamma}) = (\underline{\gamma}(\underline{x}, \bar{x}; 1), \bar{\gamma}(\underline{x}, \bar{x}; 1))$ and $(\underline{\gamma}', \bar{\gamma}') = (\underline{\gamma}'(\underline{x}, \bar{x}; 2), \bar{\gamma}'(\underline{x}, \bar{x}; 2))$ fulfill (Equation 2.33). $[\underline{x}^1, \bar{x}^1]$ is initialized to the result from the first scenario, $[(-1.72, -3.40), (1.72, 3.40)]$, shown in green (outermost solid), and the subsequent steps in the sequence are plotted in black (dashed), with the final converged hyperrectangle shown in blue (innermost solid). The resulting hyperrectangle, $[(-1.00, -1.00), (1.00, 1.00)]$ is an equilibrium of (Equation 2.82) and is thus forward invariant for (Equation 2.78) per Theorem 4.</p>	35
2.6	<p>The planar multirotor model has horizontal position y, vertical position z, and roll angle θ. The inputs are thrust u_1 in the direction perpendicular to the line segment connecting the rotors and roll angle velocity u_2. The five dimensional state x consists of y, z, θ, and the derivatives $v_y = \dot{y}, v_z = \dot{z}$, so that $x = [y \ v_y \ z \ v_z \ \theta]^T$.</p>	35
2.7	<p>The planar multirotor operates in an unknown wind field with several obstacles and is initialized with some observations (2nd row) of the wind's behavior (1st row, arrows). Using Theorem 6, we identify two invariant sets for the closed-loop safety controller (1st row, blue solid and dashed rectangles). However, only the smaller (blue solid) set can be used initially as a safety set because the larger set (blue dashed) intersects obstacles. Additionally, the largest acceptable set (1st row, green dashed) produced by Theorem 3 is more restrictive. After collecting several rounds of observations (4th row), the multirotor can now safely explore near the obstacles without collision, as denoted by the new safety set (3rd row, solid blue). Again, note that set produced by Theorem 3 (3rd row, green dashed) is more restrictive.</p>	38
3.1	<p>Execution of the Autonomous Vehicle case study from [43]. The shaded region is the portion of the road unaffected by ice. The control strategy given by algorithm 1 ensures that the vehicle always has a path back to the safety set until it can find a path to the goal.</p>	50
3.2	<p>Execution of the Planar Multirotor case study. The arrows denote the unknown wind force acting on the system.</p>	52
3.3	<p>Execution of the river motorboat case study. The arrows denote the river flow acting on the system. Additionally, the set of states where $x < -2$ and $x > 12$ are considered part of $\mathcal{X}_{\text{unsafe}}$.</p>	55

- 4.1 An illustrative example system that fits the problem setting. A planar multirotor must fly to the goal region (green) while avoiding obstacles in midair (red). There is also a wind force acting on the multirotor which varies based on its location and is unknown a priori. Observations of this force can be collected, and the objective is to guarantee a safe path to the goal, potentially while minimizing the energy spent. 61

- 4.2 For systems in which there is a cost to be minimized (i.e., energy consumption), such as the planar multirotor operating in an unknown wind field shown above, the disturbance behavior in unexplored areas of the state space may incur a lower overall cost. As shown, the system tries to reach the goal region (green rectangle) while minimizing the energy spent. The multirotor only has a few observations of the wind around its starting location, thus considering the worst-case behavior of the disturbance (Pessimism) results in the red trajectory. However, allowing the multirotor to adjust the estimated worst-case bounds (Optimism) allows the multirotor to explore, resulting in the blue trajectory. In this case, it is advantageous to be Optimistic, as the blue trajectory is closer to the calculated optimal trajectory in black. These trajectories were generated from an execution of the second case study in section 4.5. 62

- 4.3 Empirical cumulative distribution function (cdf) of the incurred cost of each strategy over 113 total runs. Each curve represents the overall results of each strategy executed in the Monte Carlo simulation. The horizontal axis denotes the cost incurred to arrive at the goal, while the vertical axis denotes the proportion of runs which incurred that cost or lower. The closer to the left a curve is, the better. For example, the Opt2 Strategy (green) incurred a cost of 150 or lower in approximately 80% of its runs. From these results, we can see that there is a tradeoff. Compared to the Pessimistic strategy (red), the Optimistic Strategies (gray) had a larger proportion of runs incur a cost of 200 or lower. However, they generally had a *smaller* proportion of runs incur a cost of 250 or lower. Only one strategy, Opt2 (green), which has a minimum probability $1 - \eta = 0.2$, unambiguously outperforms Pessimism. The respective minimum probabilities $1 - \eta$ of each strategy are outlined in Table 4.1. 72

- 5.1 The kinematic bicycle model has positions X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(\delta_f)$, front and rear distances from center of mass l_f and l_r , and inputs acceleration u_1 and steering angle $u_2 = \delta_f$ 82

- 5.2 Resulting control behaviors in the Monte Carlo simulation, showcasing the estimated reachable sets for each synthesized control strategy from solving (Equation 5.8). With only one level of information, the Conservative and Reckless strategies avoid all overlap with the obstacle. In the Conservative case, this results in a loss of performance, while in the Reckless case, this results in a loss of safety. By contrast, the Full strategy allows for some overlap between the worst-case hyperrectangles and the obstacle, preserving safety while still achieving high performance. 84

5.3	Instantiation of the MPC Case Study, where the value of μ is outside the bounds that the Reckless strategy utilizes, causing its reachable set overapproximations to be incorrect and producing erratic behavior. The Conservative strategy has a smooth path, but takes longer than it needs to due to the amount of overapproximation. The Full strategy performs large movements akin to the Reckless strategy and maintains the correct predictions of the Conservative strategy.	86
6.1	The planar multirotor model has horizontal position y , vertical position z , and roll angle θ . The inputs are thrust u_1 in the direction perpendicular to the line segment connecting the rotors and roll angle acceleration u_2 . The modified five dimensional state x consists of y, z, θ , and the velocity in the horizontal and vertical directions of the frame of the multirotor h and v , so that $x = [y \ h \ z \ v \ \theta]^T$, which fulfills Assumption 9.	95
6.2	The planar multirotor landing simulation. The multirotor attempts to make a safe landing by following the calculated reference trajectory (top, black dashed), which is a solution to (Equation 6.1) assuming the wind behaves according to the current estimated mean (bottom, blue dashed) based on the available observations (bottom, points) of the true wind behavior (bottom, black, and top, arrows). We then derive the forward invariant tube (top, red and blue) around the reference trajectory, which assumes the worst-case wind behavior, calculated by the current confidence bounds (bottom, shaded) on said behavior. At the point in the trajectory where the invariant tube deviates from the reference by a certain threshold, a recalculation of the reference trajectory and invariant tube is triggered. This process repeats until the multirotor makes a safe landing.	98
7.1	Demonstration of the advantages of including first-order interactions of the disturbance in the forward invariant tube calculation. By accounting for these interactions, the tube remains tight around the reference trajectory.	108
7.2	The planar multirotor model has horizontal position y , vertical position z , and roll angle θ . The inputs are thrust u_1 in the direction perpendicular to the line segment connecting the rotors and roll angle acceleration u_2	108
7.3	The planar multirotor trajectory tracking simulation. The multirotor attempts to track a landing reference trajectory (left, dashed), while the runtime assurance algorithm calculates a forward invariant tube (left, red and blue) using observations (right, dots, largest most recent) to estimate the mean and confidence bounds (right, dark and light blue respectively) on the disturbance (right, solid black). Per algorithm 5, an observation is collected and the trajectories are recomputed whenever the forward invariant tube deviates from the reference beyond the safety threshold. .	110

7.4 The planar multirotor Implicit ASIF simulation. The multirotor attempts to land via an uncertified controller. At all times, the filter calculates the least intrusive modification to the desired input such that a backup trajectory (left, red and blue) back to the safety region (left, green) always exists. The formulation incorporates observations (right, dots, largest most recent) to estimate the mean and confidence bounds (right, dark and light blue respectively) on the disturbance (right, solid black), which are collected whenever no part of the forward invariant tube fully fits within the safety region. 112

LIST OF ACRONYMS

GP Gaussian Process

MPC Model Predictive Control

SUMMARY

Autonomous systems operating in real-world conditions often have to contend with environmental disturbance behavior that is unknown a priori. We present a method for efficiently computing reachable sets for continuous-time systems with partially unknown dynamics. Our main assumption is that, given any hyperrectangle of states, lower and upper bounds for the unknown components are available. With this assumption, the theory of mixed monotone systems allows us to formulate an efficient method for computing hyperrectangular overapproximations of the reachable sets of the system.

We apply this formulation to a system navigating towards a goal region while avoiding unsafe regions. We derive a model predictive control scheme that avoids the unsafe region and ensures the system is always within reach of an a priori guaranteed safe region, thus ensuring feasibility until the goal is reachable. We explore this formulation further by considering multiple probability levels to increase performance.

We also consider the problem of tracking a reference trajectory for these systems. We modify the embedding system such that a single controlled trajectory corresponds to a controlled forward invariant interval tube around the reference, and utilize it in a runtime assurance mechanism that guarantees tracking of the reference trajectory within a desired threshold.

CHAPTER 1

SAFE CONTROL OF PARTIALLY UNKNOWN SYSTEMS LEVERAGING EFFICIENT REACHABILITY: AN INTRODUCTION

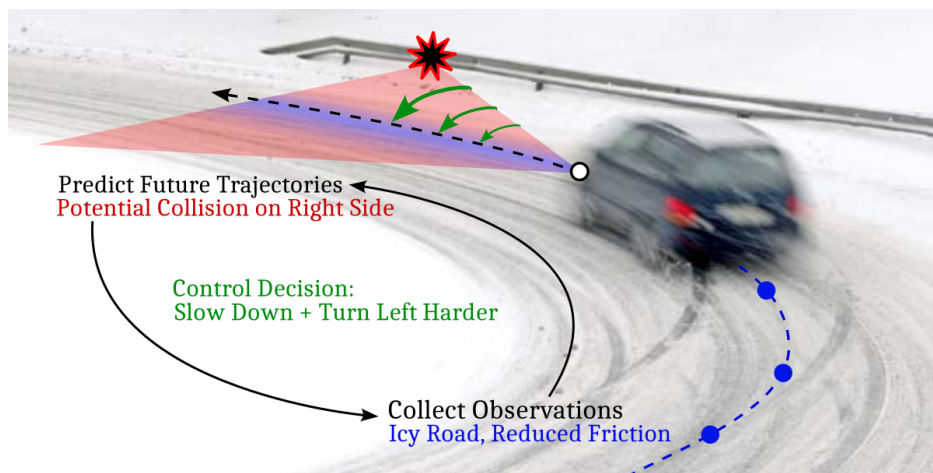


Figure 1.1: Systems with partially unknown dynamics, and strategies to enable safe operation.

Autonomous systems are ever-increasingly prevalent in modern society, from robotic fulfillment centers to autonomous vehicles. Such systems operate within and interface with real-world environments and thus require safe operation so as not to damage themselves or anything around them. Real-world conditions, however, often contain environmental disturbance behavior that is unknown a priori. This disturbance behavior may potentially affect the ability of the system to execute its planned actions, or it may cause actions that were deemed to be safe to become unsafe. We consider such systems to be *partially unknown*, and the effects of the unknown disturbance behavior must be accounted for when controlling these systems to ensure safety.

Reachability-based methods are a common approach to solving these problems. They account for all possible future trajectories of a system, and thus, it is relatively simple to define safety as ensuring no future trajectory is unsafe. However, such methods come with challenges. Common reachability techniques suffer heavily from the curse of dimensionality and are thus rendered too computationally complex for practical use. Additionally, such techniques often place assumptions

on the nature of how the unknown disturbances enter the dynamics, or even on the structure of the dynamics themselves, limiting their generalizability and occasionally introducing excess conservatism. Thus, there is a need to develop techniques that are computationally efficient, scalable, and generalizable.

The rest of this dissertation outlines the research undertaken to address these challenges. In chapter 2, we utilize the mixed monotonicity property of dynamical systems to develop a computationally efficient formulation for overapproximating the reachable sets of any nonlinear system, assuming known bounds on the disturbance behavior, and then formulate bounds that hold with high probability via Gaussian Process (GP) theory that can be directly inserted into the mixed monotone formulation. We show that the resulting reachable set overapproximations inherit the probabilistic properties of the bounds. Additionally, as observations of the disturbance behavior are collected, the bounds can be updated to reflect the current knowledge, thereby producing a tighter reachable set. As demonstrated in later chapters, this enables ongoing adaptation of control strategies as more accurate predictions of the future system behavior are obtained. This is in contrast to much of the literature on robust control, which does not inherently provide mechanisms for adjusting the effects of the disturbance as new information is gathered. Additionally, in contrast to adaptive control techniques, the utilization of GPs enables a non-parametric approach to accounting for disturbance behavior.

We then explore the reach-avoid problem for partially unknown systems in chapter 3, developing a Model Predictive Control (MPC) formulation that ensures there always exists a control strategy back to safety until enough observations of the unknown disturbance behavior are gathered and the system is able to drive to the goal with high probability. We further expand upon this formulation in chapter 4, by allowing the MPC formulation to modify the probabilistic bounds such that it considers alternative, potentially lower-cost paths to the goal, and show that doing so enables lower-cost operation. In chapter 5, we consider an alternative strategy of overapproximating multiple probability levels of disturbance behavior, enabling control strategies that are both performant and safe.

Finally, we explore assurance of partially unknown systems at runtime. In chapter 6, we modify the mixed monotonicity formulation to produce a forward invariant tube around a reference trajectory, and develop a runtime assurance algorithm that guarantees adherence to that reference trajectory below a desired safety threshold. In chapter 7, we implement time-varying behavior into the estimated bounds. We also relax an assumption made in the previous chapter, generalizing the formulation to all nonlinear systems given a known feedback control strategy, and show that it is computationally efficient enough to run in an Implicit Active Set Invariance Filtering setup.

CHAPTER 2

ESTIMATING HIGH PROBABILITY REACHABLE SETS USING GAUSSIAN PROCESSES

2.1 Introduction

The calculation of reachable or forward invariant sets for a dynamical system is often a key component of safety verification. However, these calculations tend to suffer from the curse of dimensionality, and the complexity can be compounded if the true dynamics of the system are not fully known due to inaccuracies in the model or the presence of external disturbances. *Mixed monotone systems* theory has recently proved effective for efficiently estimating rectangular forward invariant sets and overapproximations of reachable sets [1], with applications to control of practical systems of around ten state dimensions [2, 3]. Further, this theory is able to accommodate unknown but bounded disturbances into this calculation [4]. We extend these ideas by leveraging GP theory to efficiently calculate high-confidence bounds on unknown components of the dynamics in order to compute reachable sets which hold with high probability. We then show that set estimates are improved as the bounds on the unknown components tighten due to, *e.g.*, measurements of the GP, which leads to several applications of this formulation in robust control and safety verification.

An n -dimensional system is mixed monotone if there exists a *decomposition function* that separates the vector field of the system into solely increasing and solely decreasing components [5, 6, 7, 8, 1]. Said decomposition function is used to construct a $2n$ -dimensional *embedding system* that is monotone with respect to a particular southeast order (see section 2.2). This allows for the application of tools from monotone systems theory to the embedding system dynamics, which yields conclusions on the reachability and safety properties of the original system. In particular, if the original system is subject to a p -dimensional disturbance input, the resulting embedding system is subject to a $2p$ -dimensional disturbance input; considering the worst-case inputs of these

disturbances allows for the efficient computation of the reachable sets of the original system [4].

Prior works on mixed monotone systems did not consider disturbances arising from unknown but state-dependent uncertainty in the dynamics, although this is a common practical scenario. To that end, GPs have been used to model unknown functions to great effect in statistics and machine learning [9], as they are able to model distributions over any continuous domain and provide confidence estimates over a given range of function values, even with few observations. One can consequently take advantage of these confidence estimates to form high-confidence bounds on the disturbance and update these bounds as more observations are gathered.

The advantages afforded by GPs, and learning methods in general, in estimating unknown functions have not gone unnoticed by the controls community; [10] provides a method for online tuning of controller parameters using GPs while fulfilling safety criteria, [11] describes a method for incorporating Reinforcement Learning using GPs into classical model reference adaptive control, [12, 13] explore the coverage control problem for estimating unknown spatial fields using GPs, and [14] derives a uniform error bound for GPs that is used to calculate safety bounds for unknown dynamical systems. In service of providing high probability safety guarantees, [15] uses Bayesian learning to obtain a distribution over the system dynamics, [16] presents a model predictive control formulation that incorporates GPs, and [17] implements reinforcement learning to model uncertainties within control barrier function and control Lyapunov function constraints. Further, a min-norm control Lyapunov function-based stabilizing controller for control affine systems with uncertain dynamics utilizing GP regression is presented in [18]. Additionally, much work has been done in leveraging Hamilton-Jacobi reachability methods for safety. The paper [19] provides a least-restrictive safety-preserving control framework based on combining these methods with GPs, and [20] synthesizes techniques to speed up computation of Hamilton-Jacobi safe sets as uncertainties are reduced.

In this chapter, we present a method for computing reachable sets for systems whose dynamics include unknown components, drawing from the previous literature on mixed monotone systems. We accomplish this by assuming bounds on the unknown components of the system and then

show that these bounds lead to the identification of reachable and forward invariant sets, resulting in computationally efficient algorithms. We note that, when GPs are used to model the unknown components, this assumption is particularly appropriate, as it enables the calculation of bounds that are correct with high probability. We also explicitly derive the theory that allows us to quantify the probability that the GP bounds hold.

The proposed approach is well suited to high dimensional systems with low dimensional uncertainty that appears as unknown components in the dynamics. This scenario often occurs in practice, for example, in mechanical systems, where uncertain forces generally affect only the velocity dynamics and might be a function of only positional state. For such systems, bounds on the low dimensional uncertainty can be efficiently evaluated using update laws for GPs and direct sampling, while the theory of mixed monotone systems accommodates the higher dimensional dynamics, leading to tractable computations. In addition, in contrast to much of the prior work using GPs in control systems, we do not assume the dynamics are affine in the unknown components, and we model systems in continuous time.

This chapter is organized as follows. Key notation is introduced in section 2.2. In section 2.3, we formally define the assumptions made and the problem to be solved. Subsequently, in section 2.4, we illustrate the key theoretical results that solve the previously defined problem, before detailing the theory that allows us to leverage GPs in section 2.5. In section 2.6, we showcase several demonstrations of these results.

2.2 Notation

Let (x, y) denote the vector concatenation of $x, y \in \mathbb{R}^n$, i.e., $(x, y) := [x^T \ y^T]^T \in \mathbb{R}^{2n}$. Additionally, \preceq denotes the componentwise vector order, i.e., $x \preceq y$ if and only if $x_i \leq y_i$ for all $i \in \{1, \dots, n\}$ where vector components are indexed via subscript.

Given $x, y \in \mathbb{R}^n$ such that $x \preceq y$, we denote the hyperrectangle defined by the endpoints x and y using the notation $[x, y] := \{z \in \mathbb{R}^n \mid x \preceq z \text{ and } z \preceq y\}$. Also, given $a = (x, y) \in \mathbb{R}^{2n}$ with $x \preceq y$, $\llbracket a \rrbracket$ denotes the hyperrectangle formed by the first and last n components of a , i.e.,

$\llbracket a \rrbracket := [x, y]$.

Finally, let \preceq_{SE} denote the *southeast order* on \mathbb{R}^{2n} defined by $(x, x') \preceq_{\text{SE}} (y, y')$ if and only if $x \preceq y$ and $y' \preceq x'$. In particular, observe that when $x \preceq x'$ and $y \preceq y'$,

$$(x, x') \preceq_{\text{SE}} (y, y') \iff [y, y'] \subseteq [x, x']. \quad (2.1)$$

2.3 Problem Formulation

Consider the continuous-time, Lipschitz dynamical system

$$\dot{x} = f(x, w) \quad (2.2)$$

where $x \in \mathbb{R}^n$ is the system state and $w \in \mathbb{R}^p$ is an unknown, state-dependent component of the dynamics so that $w_i = g_i(x)$ where g_i is unknown. For example, g_i might account for higher order nonlinearities not explicitly captured in the model.

We make the following two fundamental assumptions throughout this document.

Assumption 1. Each w_i , $i \in \{1, \dots, p\}$ in (Equation 2.2) is *state-dependent* so that $w_i = g_i(x)$ for some unknown, Lipschitz continuous $g_i : \mathbb{R}^n \mapsto \mathbb{R}$.

Further, there exist known, Lipschitz continuous functions $\underline{\gamma}_i(x, \hat{x})$ and $\bar{\gamma}_i(x, \hat{x})$, $\underline{\gamma}_i, \bar{\gamma}_i : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$, for all $i \in \{1, \dots, p\}$ such that

$$\underline{\gamma}_i(\underline{x}, \bar{x}) \leq g_i(x) \leq \bar{\gamma}_i(\underline{x}, \bar{x}) \quad \forall x \in [\underline{x}, \bar{x}] \quad (2.3)$$

for all $\underline{x}, \bar{x} \in \mathbb{R}^n$ with $\underline{x} \preceq \bar{x}$. Without loss of generality, we assume $\underline{\gamma}_i$ and $\bar{\gamma}_i$ satisfy the natural inclusion property that for all $\underline{x} \preceq \underline{y} \preceq \bar{y} \preceq \bar{x}$ it holds that $\underline{\gamma}_i(\underline{x}, \bar{x}) \leq \underline{\gamma}_i(\underline{y}, \bar{y}) \leq \bar{\gamma}_i(\underline{y}, \bar{y}) \leq \bar{\gamma}_i(\underline{x}, \bar{x})$.

In section 2.5, we present a method for computing functions $\underline{\gamma}_i(x, \hat{x})$ and $\bar{\gamma}_i(x, \hat{x})$ satisfying Assumption 1 with high probability when g_i is modeled as a GP, and we explicitly quantify the

probability that (Equation 2.3) holds.

We denote by $g(x)$, $\underline{\gamma}$, and $\bar{\gamma}$ the vector concatenation of g_i , $\underline{\gamma}_i$, and $\bar{\gamma}_i$ for $i = 1, \dots, p$.

Assumption 2. *The system (Equation 2.2) is mixed monotone with respect to a Lipschitz decomposition function $\delta(x, w, \hat{x}, \hat{w})$, that is, $\delta : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^n \times \mathbb{R}^p \mapsto \mathbb{R}^n$ satisfies:*

1. *For all x and all w , $\delta(x, w, x, w) = f(x, w)$;*
2. *For all $i, j \in \{1, \dots, n\}$, $i \neq j$, $\frac{\partial \delta_i}{\partial x_j}(x, w, \hat{x}, \hat{w}) \geq 0$ for all x, \hat{x} and all w, \hat{w} wherever the derivative exists;*
3. *For all $i, j \in \{1, \dots, n\}$, $\frac{\partial \delta_i}{\partial \hat{x}_j}(x, w, \hat{x}, \hat{w}) \leq 0$ for all x, \hat{x} and all w, \hat{w} wherever the derivative exists; and,*
4. *For all $i \in \{1, \dots, n\}$ and all $k \in \{1, \dots, p\}$, $\frac{\partial \delta_i}{\partial w_k}(x, w, \hat{x}, \hat{w}) \geq 0$ and $\frac{\partial \delta_i}{\partial \hat{w}_k}(x, w, \hat{x}, \hat{w}) \leq 0$ for all x, \hat{x} and all w, \hat{w} wherever the derivative exists.*

Assumption 2 is not particularly restrictive as it has been shown in [21] that, for any system, some decomposition function δ exists that satisfies the above conditions, though it may or may not be readily available in closed-form. Generally, domain-specific knowledge is leveraged to obtain closed-form decomposition functions, and for some classes of systems such as those for which the partial derivatives of f are bounded, decomposition functions are readily constructed from f . Additionally, the choice of decomposition function affects the conservatism of the reachable set overapproximation. For examples of decomposition functions, see the case studies outlined in [22], or [1] and citations therein for more details. Lastly, we note that for simplicity, we take \mathbb{R}^n as the domain, although it is possible to restrict to a domain that is a hyperrectangular subset of \mathbb{R}^n , as described in [1].

For initial condition $x_0 \in \mathbb{R}^n$, let $\phi(t, x_0)$ denote the resulting state trajectory of (Equation 2.2) when $w = g(x)$, that is, $\phi(t, x_0)$ satisfies $\frac{d}{dt}\phi(t, x_0) = f(\phi(t, x_0), g(\phi(t, x_0)))$. Given some $X_0 \subseteq \mathbb{R}^n$, the T -horizon reachable set from X_0 for (Equation 2.2) is the set of states reachable over the

time horizon T from any initial condition $x_0 \in X_0$ and is denoted

$$R(T, X_0) = \{\phi(T, x_0) \mid x_0 \in X_0\}. \quad (2.4)$$

Even when the dynamics are fully known, computing exact reachable sets is generally not possible. Thus, we are interested in computing approximations of reachable sets. In particular, over-approximations are often preferred for, e.g., safety verification. A key feature of mixed monotone systems is that hyperrectangular over-approximations of reachable sets are efficiently computed from trajectories of a $2n$ dimensional embedding system constructed from the decomposition function δ . In section 2.4, we extend this fundamental property to systems with state-dependent uncertainty satisfying Assumption 1, expanding on our results from [23]. The focus of this chapter is on the tractable computation of over-approximations for reachable sets of (Equation 2.2).

Problem 1. *Given $X_0 = [\underline{x}_0, \bar{x}_0]$ for some $\underline{x}_0, \bar{x}_0 \in \mathbb{R}^n$ with $\underline{x}_0 \preceq \bar{x}_0$, compute $\widehat{R}(T, X_0)$ such that $R(T, X_0) \subseteq \widehat{R}(T, X_0)$.*

Closely related to the problem of computing reachable sets is the problem of computing forward invariant sets for (Equation 2.2).

Problem 2. *Identify sets $A \subseteq \mathbb{R}^n$ such that $R(t, A) \subseteq A$ for all $t \geq 0$.*

In the next sections, we present a solution to Problem 1 and Problem 2 using mixed monotone systems theory. We also use the theory of GPs to derive a general approach to satisfying Assumption 1 and, in particular, obtaining the necessary functions $\underline{\gamma}$ and $\bar{\gamma}$ such that (Equation 2.3), and consequently the identified reachable sets, holds with high probability.

2.4 High Probability Reachable Sets

We present solutions to Problem 1 and Problem 2, including results that show improvements to the tightness of the calculated sets as bounds on the unknown components are tightened. We then

outline an additional refinement to our formulation that results in more accurate reachable and forward invariant sets at the cost of increased computational complexity.

We begin by recalling the fundamental result of mixed monotone systems theory (see, *e.g.*, [1]). Construct the *embedding system* with state $(x, \hat{x}) \in \mathbb{R}^n \times \mathbb{R}^n$ and disturbance $(w, \hat{w}) \in \mathbb{R}^p \times \mathbb{R}^p$

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \varepsilon(x, w, \hat{x}, \hat{w}) := \begin{bmatrix} \delta(x, w, \hat{x}, \hat{w}) \\ \delta(\hat{x}, \hat{w}, x, w) \end{bmatrix}. \quad (2.5)$$

Denote the state of (Equation 2.5) at time t when initialized at $(\underline{x}_0, \bar{x}_0)$ under some disturbance input signal $(w, \hat{w}) : [0, \infty) \rightarrow \mathbb{R}^p \times \mathbb{R}^p$ by $\Phi^\varepsilon(t; (\underline{x}_0, \bar{x}_0), (w, \hat{w}))$. The fundamental result of mixed monotone systems theory is that (Equation 2.5) is a monotone control system as defined in [24] with respect to the southeast order on state and disturbance; that is, given $a, a' \in \mathbb{R}^n \times \mathbb{R}^n$ and $b, b' : [0, \infty) \rightarrow \mathbb{R}^p \times \mathbb{R}^p$ such that $a \preceq_{\text{SE}} a'$ and $b(t) \preceq_{\text{SE}} b'(t)$ for all $t \geq 0$, then for all $t \geq 0$,

$$\Phi^\varepsilon(t; a, b) \preceq_{\text{SE}} \Phi^\varepsilon(t; a', b'). \quad (2.6)$$

2.4.1 Reachable Sets

Our main result for calculating reachable sets is as follows.

Theorem 1. *Consider (Equation 2.2) satisfying Assumption 1 and Assumption 2. Let $X_0 = [\underline{x}_0, \bar{x}_0]$ for $\underline{x}_0, \bar{x}_0 \in \mathbb{R}^n$, with $\underline{x}_0 \preceq \bar{x}_0$, be a hyperrectangular set of initial conditions. Let $(x(t), \hat{x}(t))$ be the solution to the $2n$ dimensional system*

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e(x, \hat{x}) := \begin{bmatrix} \delta(x, \underline{\gamma}(x, \hat{x}), \hat{x}, \bar{\gamma}(x, \hat{x})) \\ \delta(\hat{x}, \bar{\gamma}(x, \hat{x}), x, \underline{\gamma}(x, \hat{x})) \end{bmatrix} \quad (2.7)$$

with initial condition $(x(0), \hat{x}(0)) = (\underline{x}_0, \bar{x}_0)$. Then, $R(T, X_0) \subseteq [x(T), \hat{x}(T)]$ for all $T \geq 0$.

Proof. We first construct the embedding system (Equation 2.5), then substitute $(\underline{\gamma}(x, \hat{x}), \bar{\gamma}(x, \hat{x}))$ for (w, \hat{w}) , giving the embedding system (Equation 2.7). We denote the solutions $(x(t), \hat{x}(t))$ of

this system by $\Phi^e(t; (\underline{x}_0, \bar{x}_0))$, which we note is equivalent to $\Phi^e(t; (\underline{x}_0, \bar{x}_0), (\underline{\gamma}, \bar{\gamma}))$, where it is understood that $(\underline{\gamma}, \bar{\gamma})$ is evaluated along the (x, \hat{x}) trajectory. Note also that

$$\Phi^e(t; (x_0, x_0), (g, g)) = (\phi(t, x_0), \phi(t, x_0)), \quad (2.8)$$

where it is understood that (g, g) is evaluated along the system trajectory, *i.e.*, $\Phi^e(t; (x_0, x_0), (g, g))$ is equivalent to two copies of state trajectories of the original dynamics (Equation 2.2).

Now, Assumption 1 equivalently states that

$$(\underline{\gamma}(x, \hat{x}), \bar{\gamma}(x, \hat{x})) \preceq_{\text{SE}} (g(x'), g(x')) \quad \forall x \preceq x' \preceq \hat{x}. \quad (2.9)$$

Moreover, for any initial state $x_0 \in [\underline{x}_0, \bar{x}_0]$, we have equivalently, $(\underline{x}_0, \bar{x}_0) \preceq_{\text{SE}} (x_0, x_0)$. Per the fundamental result of mixed monotone systems theory (Equation 2.6), (Equation 2.9) implies

$$\Phi^e(t; (\underline{x}_0, \bar{x}_0)) \preceq_{\text{SE}} \Phi^e(t; (x_0, x_0), (g, g)) \quad (2.10)$$

for all $t \geq 0$. By (Equation 2.8),

$$\phi(t, x_0) \in \llbracket \Phi^e(t; (\underline{x}_0, \bar{x}_0)) \rrbracket \quad (2.11)$$

for all $t \geq 0$. Finally, recalling (Equation 2.4) gives us

$$R(T, X_0) \subseteq \llbracket \Phi^e(T; (\underline{x}_0, \bar{x}_0)) \rrbracket = [x(T), \hat{x}(T)]. \quad (2.12)$$

■

Theorem 1 establishes that $\widehat{R}(T, X_0) := [x(T), \hat{x}(T)]$, where $x(T), \hat{x}(T)$ are obtained as the solution to the ODE (Equation 2.7), provides a hyperrectangular over-approximation of the true reachable set $R(T, X_0)$, thus solving Problem 1.

We show in section 2.5 that the theory of GPs naturally leads to a methodology for modeling

and obtaining $\underline{\gamma}$ and $\bar{\gamma}$ satisfying Assumption 2. In this case, it is further reasonable to envision updating the bounds $\underline{\gamma}$ and $\bar{\gamma}$ by incorporating newly collected data into estimation of the GPs. In this case, estimates of reachable sets become tighter, provided that the updated bounds $\underline{\gamma}$ and $\bar{\gamma}$ become tighter. This is formalized in Theorem 2.

Theorem 2. Consider (Equation 2.2) satisfying Assumption 2. Let $\underline{\gamma}, \bar{\gamma}$ be a pair of bounds satisfying Assumption 1, and let $\underline{\gamma}', \bar{\gamma}'$ be another pair of bounds satisfying Assumption 1. Suppose further that the bounds $\underline{\gamma}', \bar{\gamma}'$ are tighter than $\underline{\gamma}, \bar{\gamma}$, that is, for all \underline{x}, \bar{x} with $\underline{x} \preceq \bar{x}$, it holds that

$$\underline{\gamma}(\underline{x}, \bar{x}) \preceq \underline{\gamma}'(\underline{x}, \bar{x}) \preceq \bar{\gamma}'(\underline{x}, \bar{x}) \preceq \bar{\gamma}(\underline{x}, \bar{x}). \quad (2.13)$$

Construct the embedding systems

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e'(x, \hat{x}) := \begin{bmatrix} \delta(x, \underline{\gamma}'(x, \hat{x}), \hat{x}, \bar{\gamma}'(x, \hat{x})) \\ \delta(\hat{x}, \bar{\gamma}'(x, \hat{x}), x, \underline{\gamma}'(x, \hat{x})) \end{bmatrix} \quad (2.14)$$

and

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e(x, \hat{x}) := \begin{bmatrix} \delta(x, \underline{\gamma}(x, \hat{x}), \hat{x}, \bar{\gamma}(x, \hat{x})) \\ \delta(\hat{x}, \bar{\gamma}(x, \hat{x}), x, \underline{\gamma}(x, \hat{x})) \end{bmatrix}. \quad (2.15)$$

Then

$$\llbracket \Phi^{e'}(T; (\underline{x}_0, \bar{x}_0)) \rrbracket \subseteq \llbracket \Phi^e(T; (\underline{x}_0, \bar{x}_0)) \rrbracket. \quad (2.16)$$

Proof. We first note that (Equation 2.13) equivalently states

$$(\underline{\gamma}(\underline{x}, \bar{x}), \bar{\gamma}(\underline{x}, \bar{x})) \preceq_{\text{SE}} (\underline{\gamma}'(\underline{x}, \bar{x}), \bar{\gamma}'(\underline{x}, \bar{x})) \quad \forall \underline{x} \preceq \bar{x}. \quad (2.17)$$

It thus follows that, per the fundamental result of mixed monotone systems theory (Equation 2.6),

$$\Phi^e(t; (x, \hat{x}), (\underline{\gamma}(x, \hat{x}), \bar{\gamma}(x, \hat{x}))) \preceq_{\text{SE}} \Phi^{e'}(t; (x, \hat{x}), (\underline{\gamma}'(x, \hat{x}), \bar{\gamma}'(x, \hat{x}))), \quad (2.18)$$

or, equivalently,

$$\Phi^e(T; (\underline{x}_0, \bar{x}_0)) \preceq_{\text{SE}} \Phi^{e'}(T; (\underline{x}_0, \bar{x}_0)), \quad (2.19)$$

which is equivalent to (Equation 2.16). ■

The first case study of section 2.6 uses Theorem 2 to learn a tight overapproximation of dynamics subject to a sudden unknown disturbance by collecting observations of the unknown component's behavior.

2.4.2 Forward Invariant Sets

The same mixed monotone formulation allows for the efficient computation of forward invariant sets.

Theorem 3. *Consider (Equation 2.2) satisfying Assumption 1 and Assumption 2. If $\exists \underline{x}^*, \bar{x}^*, \underline{w}^*, \bar{w}^*$ with $\underline{x}^* \preceq \bar{x}^*$ and $\underline{w}^* \preceq \bar{w}^*$ such that*

$$\underline{w}^* \preceq \underline{\gamma}(\underline{x}^*, \bar{x}^*) \text{ and } \bar{\gamma}(\underline{x}^*, \bar{x}^*) \preceq \bar{w}^* \quad (2.20)$$

and

$$\delta(\underline{x}^*, \underline{w}^*, \bar{x}^*, \bar{w}^*) \succeq 0 \text{ and } \delta(\bar{x}^*, \bar{w}^*, \underline{x}^*, \underline{w}^*) \preceq 0, \quad (2.21)$$

then $[\underline{x}^*, \bar{x}^*]$ is forward invariant for (Equation 2.2).

Proof. We construct an embedding system similar to the system outlined by (Equation 2.7), substituting $(\underline{w}^*, \bar{w}^*)$ for $(\underline{\gamma}(x, \hat{x}), \bar{\gamma}(x, \hat{x}))$:

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e^*(x, \hat{x}) := \begin{bmatrix} \delta(x, \underline{w}^*, \hat{x}, \bar{w}^*) \\ \delta(\hat{x}, \bar{w}^*, x, \underline{w}^*) \end{bmatrix}. \quad (2.22)$$

The *upper triangle* of this embedding system is defined as $\mathcal{T} := \{(x, \hat{x}) \in \mathbb{R}^n \times \mathbb{R}^n \mid x \preceq \hat{x}\}$, and the set of points in \mathcal{T} such that the vector field of the embedding system points into the southeast

cone is defined as $\mathcal{S} := \{(x, \hat{x}) \in \mathcal{T} \mid 0 \preceq_{\text{SE}} e^*(x, \hat{x})\}$. The set \mathcal{T} is called the upper triangle as it can be visualized as the set of states above the $x = \hat{x}$ line in the sense of the partial order \preceq . Per [4, Lemma 1], \mathcal{T} is forward invariant for (Equation 2.22). Further, as a direct result of [25, Ch. 3, Prop 2.1], \mathcal{S} is also forward invariant for (Equation 2.22) and $\Phi^{e^*}(t_1; a) \preceq_{\text{SE}} \Phi^{e^*}(t_2; a)$ for all $a \in \mathcal{S}$ and all $0 \leq t_1 \leq t_2$. In particular, $a \preceq_{\text{SE}} \Phi^{e^*}(t; a)$ for all $a \in \mathcal{S}$ and all $t \geq 0$, i.e., by (Equation 2.1), $[\Phi^{e^*}(t; a)] \subseteq [a]$ for all $a \in \mathcal{S}$ and all $t \geq 0$.

Let $a^* := (\underline{x}^*, \bar{x}^*)$. By (Equation 2.21), $a^* \in \mathcal{S}$, so that $[\Phi^{e^*}(t; a^*)] \subseteq [a^*]$ for all $t \geq 0$. From Theorem 1, for any $t \geq 0$, we have that $R(t, [a^*]) \subseteq [\Phi^e(t; a^*)]$ where Φ^e is as defined in the proof of Theorem 1. Next, (Equation 2.20) and Theorem 2 implies $[\Phi^e(t; a^*)] \subseteq [\Phi^{e^*}(t; a^*)]$. Combined, we have $R(t, [a^*]) \subseteq [\Phi^e(t; a^*)] \subseteq [\Phi^{e^*}(t; a^*)] \subseteq [a^*]$ for all $t \geq 0$, i.e., $[a^*] = [\underline{x}^*, \bar{x}^*]$ is forward invariant for (Equation 2.2). ■

We immediately have the following corollary.

Corollary 1. *Consider (Equation 2.2) satisfying Assumption 1 and Assumption 2. If $(\underline{x}^*, \bar{x}^*)$ is an equilibrium for the embedding system (Equation 2.7), then $[\underline{x}^*, \bar{x}^*]$ is forward invariant for (Equation 2.2).*

Corollary 1 suggests a numerical method for identifying invariant sets for (Equation 2.2): compute equilibria of the embedding system (Equation 2.7) by, e.g., initializing the embedding system dynamics at some point and simulating the dynamics to determine if the trajectory converges. One difficulty of this method, however, is that the functions $\underline{\gamma}$ and $\bar{\gamma}$ must be evaluated at each point along the entire trajectory. As discussed in the next section, this is computationally reasonable in some cases, but may be impractical in other cases.

Instead, Theorem 3 offers an alternative method for identifying invariant sets that relies on evaluating $\underline{\gamma}$ and $\bar{\gamma}$ at a sequence of points and simulating the resulting embedding dynamics with $\underline{\gamma}$ and $\bar{\gamma}$ fixed to these evaluations.

To that end, consider an initial $\underline{x}^1 \preceq \bar{x}^1$ and construct the sequences $\{\underline{x}^k\}_{k=1}^{\infty}$, $\{\bar{x}^k\}_{k=1}^{\infty}$ accord-

ing to the recursion

$$\underline{w}^k := \underline{\gamma}(\underline{x}^k, \bar{x}^k), \bar{w}^k := \bar{\gamma}(\underline{x}^k, \bar{x}^k) \quad (2.23)$$

$$(\underline{x}^{k+1}, \bar{x}^{k+1}) := \lim_{t \rightarrow \infty} (x^k(t), \hat{x}^k(t)) \quad (2.24)$$

provided the limit exists, where $(x^k(t), \hat{x}^k(t))$ solves

$$\dot{x}^k = \delta(x^k, \underline{w}^k, \hat{x}^k, \bar{w}^k), \dot{\hat{x}}^k = \delta(\hat{x}^k, \bar{w}^k, x^k, \underline{w}^k) \quad (2.25)$$

with initial condition $x^k(0) = \underline{x}^k, \hat{x}^k(0) = \bar{x}^k$.

Theorem 4. *Consider (Equation 2.2) satisfying Assumption 1 and Assumption 2. For any initial $\underline{x}^1 \preceq \bar{x}^1$, if the sequences $\{\underline{x}^k\}_{k=1}^{\infty}, \{\bar{x}^k\}_{k=1}^{\infty}$ constructed as per (Equation 2.23)–(Equation 2.25) are well-defined (i.e. the limits exist) and converge, then $(\underline{x}^*, \bar{x}^*) := \lim_{k \rightarrow \infty} (\underline{x}^k, \bar{x}^k)$ constitutes an equilibrium of (Equation 2.7) and, hence, $[\underline{x}^*, \bar{x}^*]$ is forward invariant for (Equation 2.2).*

In addition, if for some K it holds that

$$\underline{x}^K \preceq \underline{x}^{K+1} \text{ and } \bar{x}^{K+1} \preceq \bar{x}^K, \quad (2.26)$$

then for all $k \geq K$, it holds that $\underline{x}^k \preceq \underline{x}^{k+1}$ and $\bar{x}^{k+1} \preceq \bar{x}^k$ and the sequences $\{\underline{x}^k\}_{k=1}^{\infty}, \{\bar{x}^k\}_{k=1}^{\infty}$ converge. Moreover, each $[\underline{x}^k, \bar{x}^k]$ for $k \geq K$ is forward invariant for (Equation 2.2).

The proof of Theorem 4 relies in part on the following lemma, which is an immediate consequence of Property 4 of the decomposition function δ stated in Assumption 2.

Lemma 1. *Consider (Equation 2.2) satisfying Assumption 1 and Assumption 2, and $\underline{x}^*, \bar{x}^*, \underline{w}^*, \bar{w}^*$ with $\underline{x}^* \preceq \bar{x}^*$ and $\underline{w}^* \preceq \bar{w}^*$ such that*

$$\delta(\underline{x}^*, \underline{w}^*, \bar{x}^*, \bar{w}^*) = 0 \text{ and } \delta(\bar{x}^*, \bar{w}^*, \underline{x}^*, \underline{w}^*) = 0. \quad (2.27)$$

If \underline{w}' and \bar{w}' satisfy $\underline{w}^* \preceq \underline{w}' \preceq \bar{w}' \preceq \bar{w}^*$, then

$$\delta(\underline{x}^*, \underline{w}', \bar{x}^*, \bar{w}') \succeq 0 \text{ and } \delta(\bar{x}^*, \bar{w}', \underline{x}^*, \underline{w}') \preceq 0. \quad (2.28)$$

Proof of Theorem 4. For the first part of Theorem 4, construct the embedding system described by (Equation 2.25) for each iteration k ,

$$\begin{bmatrix} \dot{x}^k \\ \dot{\hat{x}}^k \end{bmatrix} = e^k(x^k, \hat{x}^k) := \begin{bmatrix} \delta(x^k, \underline{w}^k, \hat{x}^k, \bar{w}^k) \\ \delta(\hat{x}^k, \bar{w}^k, x^k, \underline{w}^k) \end{bmatrix}. \quad (2.29)$$

Since the limit $\lim_{t \rightarrow \infty} (x^k(t), \hat{x}^k(t))$ exists, $(\underline{x}^{k+1}, \bar{x}^{k+1})$ as defined by (Equation 2.24) must be such that

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e^k(\underline{x}^{k+1}, \bar{x}^{k+1}) = 0, \quad (2.30)$$

i.e., $(\underline{x}^{k+1}, \bar{x}^{k+1})$ is an equilibrium for e^k .

Because the sequences $\{\underline{x}^k\}_{k=1}^\infty$, $\{\bar{x}^k\}_{k=1}^\infty$ converge, and because δ is Lipschitz continuous by Assumption 2, the Lipschitz continuous function $e^*(x, \hat{x}) = \begin{bmatrix} \delta(x, \underline{w}^*, \hat{x}, \bar{w}^*) & \delta(\hat{x}, \bar{w}^*, x, \underline{w}^*) \end{bmatrix}^T$ where $(\underline{w}^*, \bar{w}^*) := \lim_{k \rightarrow \infty} (\underline{w}^k, \bar{w}^k)$ satisfies $e^*(\underline{x}^*, \bar{x}^*) = 0$ where $(\underline{x}^*, \bar{x}^*) := \lim_{k \rightarrow \infty} (\underline{x}^k, \bar{x}^k)$. In turn, this means that $(\underline{x}^*, \bar{x}^*)$ is an equilibrium of the embedding system $\begin{bmatrix} \dot{x} & \dot{\hat{x}} \end{bmatrix}^T = e^*(x, \hat{x})$, which we recall from Corollary 1 means that $[\underline{x}^*, \bar{x}^*]$ is forward invariant for (Equation 2.2).

For the second part of Theorem 4, we note from the inclusion property in Assumption 2 that, for the K at which (Equation 2.26) holds, it must be true that $\underline{\gamma}(\underline{x}^K, \bar{x}^K) \preceq \underline{\gamma}(\underline{x}^{K+1}, \bar{x}^{K+1}) \preceq \bar{\gamma}(\underline{x}^{K+1}, \bar{x}^{K+1}) \preceq \bar{\gamma}(\underline{x}^K, \bar{x}^K)$, or, equivalently, $\underline{w}^K \preceq \underline{w}^{K+1} \preceq \bar{w}^{K+1} \preceq \bar{w}^K$. From (Equation 2.30), it also holds that $\delta(\underline{x}^{K+1}, \underline{w}^K, \bar{x}^{K+1}, \bar{w}^K) = 0$ and $\delta(\bar{x}^{K+1}, \bar{w}^K, \underline{x}^{K+1}, \underline{w}^K) = 0$. Thus, by Lemma 1,

$$\delta(\underline{x}^{K+1}, \underline{w}^{K+1}, \bar{x}^{K+1}, \bar{w}^{K+1}) \succeq 0, \quad (2.31)$$

$$\delta(\bar{x}^{K+1}, \bar{w}^{K+1}, \underline{x}^{K+1}, \underline{w}^{K+1}) \preceq 0, \quad (2.32)$$

which subsequently means that $\underline{x}^{K+1} \preceq \underline{x}^{K+2}$ and $\bar{x}^{K+2} \preceq \bar{x}^{K+1}$ must also hold, which means

that $\underline{w}^{K+1} \preceq \underline{w}^{K+2} \preceq \overline{w}^{K+2} \preceq \overline{w}^{K+1}$ also holds, and so on by induction. This also extends back one step to iteration K , i.e., it must also have been true that $\delta(\underline{x}^K, \underline{w}^K, \overline{x}^K, \overline{w}^K) \succeq 0$ and $\delta(\overline{x}^K, \overline{w}^K, \underline{x}^K, \underline{w}^K) \preceq 0$ in order for (Equation 2.26) to hold in the first place (though we note that $\underline{w}^{K-1} \preceq \underline{w}^K \preceq \overline{w}^K \preceq \overline{w}^{K-1}$ does not necessarily have to hold for this to be the case).

As a result, for all $k \geq K$, it holds that $\underline{x}^k \preceq \underline{x}^{k+1}$ and $\overline{x}^{k+1} \preceq \overline{x}^k$ and the sequences $\{\underline{x}^k\}_{k=1}^\infty$, $\{\overline{x}^k\}_{k=1}^\infty$ converge. Additionally, as $\delta(\underline{x}^k, \underline{w}^k, \overline{x}^k, \overline{w}^k) \succeq 0$ and $\delta(\overline{x}^k, \overline{w}^k, \underline{x}^k, \underline{w}^k) \preceq 0$ hold for all $k \geq K$, by Theorem 3 each $[\underline{x}^k, \overline{x}^k]$ for $k \geq K$ is forward invariant for (Equation 2.2). ■

Theorem 3 and Lemma 1 lead to another consequence regarding identifying forward invariant sets. As previously established, it is reasonable to envision updating the bounds $\underline{\gamma}$ and $\overline{\gamma}$ by incorporating newly collected data into estimation of the GPs. As a result, estimates of any forward invariant sets can be used to establish new, tighter forward invariant sets, provided that the updated bounds $\underline{\gamma}$ and $\overline{\gamma}$ become tighter. This is formalized in Proposition 1 below.

Proposition 1. *Consider (Equation 2.2) satisfying Assumption 2. Let $\underline{\gamma}, \overline{\gamma}$ be one pair of bounds satisfying Assumption 1, and let $\underline{\gamma}', \overline{\gamma}'$ be another pair of bounds satisfying Assumption 1. Suppose further that the bounds $\underline{\gamma}, \overline{\gamma}$ are tighter than $\underline{\gamma}', \overline{\gamma}'$, that is, for all $\underline{x}, \overline{x}$ with $\underline{x} \preceq \overline{x}$, it holds that*

$$\underline{\gamma}'(\underline{x}, \overline{x}) \preceq \underline{\gamma}(\underline{x}, \overline{x}) \preceq \overline{\gamma}(\underline{x}, \overline{x}) \preceq \overline{\gamma}'(\underline{x}, \overline{x}). \quad (2.33)$$

Let $\underline{x}^*, \overline{x}^*$ satisfy

$$0 = \delta(\underline{x}^*, \underline{\gamma}'(\underline{x}^*, \overline{x}^*), \overline{x}^*, \overline{\gamma}'(\underline{x}^*, \overline{x}^*)) \quad (2.34)$$

$$0 = \delta(\overline{x}^*, \overline{\gamma}'(\underline{x}^*, \overline{x}^*), \underline{x}^*, \underline{\gamma}'(\underline{x}^*, \overline{x}^*)), \quad (2.35)$$

that is, $\underline{x}^*, \overline{x}^*$ is an equilibrium for the embedding dynamics constructed using $\underline{\gamma}'$ and $\overline{\gamma}'$. Construct the sequences $\{\underline{x}^k\}_{k=1}^\infty$, $\{\overline{x}^k\}_{k=1}^\infty$ as prescribed in (Equation 2.23)–(Equation 2.25) with ini-

tial $\underline{x}^1 = \underline{x}^{*'} and \bar{x}^1 = \bar{x}^{*'}. Then$

$$\underline{x}^k \preceq \underline{x}^{k+1} \text{ and } \bar{x}^{k+1} \preceq \bar{x}^k \quad (2.36)$$

for all $k \geq 1$ so that, in particular, the conclusions of Theorem 4 hold for all $k \geq 1$, namely, the sequences $\{\underline{x}^k\}_{k=1}^\infty, \{\bar{x}^k\}_{k=1}^\infty$ converge and each $[\underline{x}^k, \bar{x}^k]$ is forward invariant for (Equation 2.2).

Proof. Given (Equation 2.33), $(\underline{x}^*, \bar{x}^*) = (\underline{x}^{*'}, \bar{x}^{*'})$, $(\underline{w}^*, \bar{w}^*) = (\underline{\gamma}', \bar{\gamma}')$, and $(\underline{w}^{*'}, \bar{w}^{*'}) = (\underline{\gamma}, \bar{\gamma})$ satisfy the conditions for Lemma 1. Thus, from Theorem 4 it holds that

$$\delta(\underline{x}^1, \underline{w}^1, \bar{x}^1, \bar{w}^1) \succeq 0, \delta(\bar{x}^1, \bar{w}^1, \underline{x}^1, \underline{w}^1) \preceq 0. \quad (2.37)$$

The rest of the proof follows from the second part of the proof of Theorem 4. ■

Theorem 3 and Corollary 1 indicate that the hyperrectangular set $A := [\underline{x}^*, \bar{x}^*]$, such that $(\underline{x}^*, \bar{x}^*)$ is an equilibrium for the embedding system (Equation 2.22), is a forward invariant set for (Equation 2.2). Additionally, Theorem 4 and Proposition 1 describe a computationally tractable algorithm for calculating $[\underline{x}^*, \bar{x}^*]$. Thus, Theorem 3, Theorem 4, Proposition 1, and Corollary 1 solve Problem 2.

2.4.3 Refinement of Reachable and Forward Invariant Sets

Since we consider systems in continuous time, we can further refine the estimation of our reachable and forward invariant sets by noting that, when calculating these sets, we are only interested in the potential behavior of the disturbance at the boundary of the set. Thus, we can modify the range over which we calculate the bounds $\underline{\gamma}, \bar{\gamma}$ in our formulations to reflect this, at the cost of increased computational complexity. This idea is formalized in our next set of theoretical results which refine Theorem 1 and Theorem 3. We use the notation $a_{[i:b]}$ to denote a vector a whose i th term has been replaced by the i th term of vector b .

Theorem 5. Consider the hypotheses of Theorem 1 and construct the system

$$\begin{bmatrix} \dot{x}' \\ \dot{\hat{x}}' \end{bmatrix} = e'(x, \hat{x}) := \begin{bmatrix} \delta_1(x, \underline{\gamma}(x, \hat{x}_{[1:x]}), \hat{x}, \overline{\gamma}(x, \hat{x}_{[1:x]})) \\ \vdots \\ \delta_n(x, \underline{\gamma}(x, \hat{x}_{[n:x]}), \hat{x}, \overline{\gamma}(x, \hat{x}_{[n:x]})) \\ \delta_1(\hat{x}, \overline{\gamma}(x_{[1:\hat{x}]}, \hat{x}), x, \underline{\gamma}(x_{[1:\hat{x}]}, \hat{x})) \\ \vdots \\ \delta_n(\hat{x}, \overline{\gamma}(x_{[n:\hat{x}]}, \hat{x}), x, \underline{\gamma}(x_{[n:\hat{x}]}, \hat{x})) \end{bmatrix} \quad (2.38)$$

as an alternative to the embedding system (Equation 2.7). Then, (Equation 2.38) is such that the conclusion of Theorem 1 remains valid using instead trajectories of (Equation 2.38), that is, $R(T, X_0) \subseteq [x'(T), \hat{x}'(T)]$, where $(x'(t), \hat{x}'(t))$ is the solution to the $2n$ -dimensional system (Equation 2.38) with initial condition $(x'(0), \hat{x}'(0)) = (\underline{x}_0, \overline{x}_0)$.

In Theorem 5, when determining a specific state's update behavior, we limit the calculation of the disturbance bounds to the relevant face of the state hyperrectangle, as disturbances outside of this face do not affect the state.

Proof of Theorem 5. Per the problem setup and Assumption 2, the system (Equation 2.2) is mixed monotone with respect to a function $\delta(x, w, \hat{x}, \hat{w})$, where the true behavior of w is dictated by an unknown function $g(x)$. Define

$$\underline{G}(\underline{x}, \overline{x}) = \max z \text{ s.t. } z \leq g(x) \forall x \in [\underline{x}, \overline{x}] \quad (2.39)$$

$$\overline{G}(\underline{x}, \overline{x}) = \min z \text{ s.t. } z \geq g(x) \forall x \in [\underline{x}, \overline{x}] \quad (2.40)$$

for $\underline{x} \preceq \overline{x}$, which results in the tightest possible inclusion function, i.e. $g(x) \in [\underline{G}(\underline{x}, \overline{x}), \overline{G}(\underline{x}, \overline{x})]$ for all $x \in [\underline{x}, \overline{x}]$.

Define a new system $\dot{x} = f'(x, W) := f(x, g(x) + W)$. Each component $f'_j(x, W)$ is mixed-

monotone with respect to the decomposition function

$$\delta'_j(x, W, \widehat{x}, \widehat{W}) := \begin{cases} \delta_j(x, \underline{G}(x, \widehat{x}_{[j:x]}) + w^j, \widehat{x}, \overline{G}(x, \widehat{x}_{[j:x]}) + \widehat{w}^j), & x \preceq \widehat{x} \\ \delta_j(\widehat{x}, \overline{G}(x_{[j:\widehat{x}]}, \widehat{x}) + \widehat{w}^j, x, \underline{G}(x_{[j:\widehat{x}]}, \widehat{x}) + w^j), & x \succeq \widehat{x} \end{cases} \quad (2.41)$$

where $W = [w^1, w^2, \dots, w^n]^T \in \mathbb{R}^{pn}$ and $\widehat{W} = [\widehat{w}^1, \widehat{w}^2, \dots, \widehat{w}^n]^T \in \mathbb{R}^{pn}$. One can verify the derivative conditions for this by utilizing the chain rule and noting the following properties of \underline{G} and \overline{G} :

$$\begin{aligned} \frac{\partial \underline{G}(x, \widehat{x}_{[i:x]})}{\partial x_j} &\geq 0, \quad \frac{\partial \overline{G}(x, \widehat{x}_{[i:x]})}{\partial x_j} \leq 0 \quad \forall_{i,j \in n, i \neq j} \\ \frac{\partial \underline{G}(x, \widehat{x}_{[i:x]})}{\partial \widehat{x}_j} &\leq 0, \quad \frac{\partial \overline{G}(x, \widehat{x}_{[i:x]})}{\partial \widehat{x}_j} \geq 0 \quad \forall_{i,j \in n} \\ \frac{\partial \underline{G}(x_{[i:\widehat{x}]}, \widehat{x})}{\partial x_j} &\geq 0, \quad \frac{\partial \overline{G}(x_{[i:\widehat{x}]}, \widehat{x})}{\partial x_j} \leq 0 \quad \forall_{i,j \in n} \\ \frac{\partial \underline{G}(x_{[i:\widehat{x}]}, \widehat{x})}{\partial \widehat{x}_j} &\leq 0, \quad \frac{\partial \overline{G}(x_{[i:\widehat{x}]}, \widehat{x})}{\partial \widehat{x}_j} \geq 0 \quad \forall_{i,j \in n, i \neq j}. \end{aligned}$$

Then, construct the embedding system

$$\begin{bmatrix} \dot{x} \\ \dot{\widehat{x}} \end{bmatrix} = e(x, \widehat{x}, W_1, \widehat{W}_1, W_2, \widehat{W}_2) = \begin{bmatrix} \delta'(x, W_1, \widehat{x}, \widehat{W}_1) \\ \delta'(\widehat{x}, \widehat{W}_2, x, W_2) \end{bmatrix} \quad (2.42)$$

which has inputs $(W_1, \widehat{W}_1, \widehat{W}_2, W_2) \in \mathbb{R}^{4pn}$. This is a modified embedding system without symmetric disturbances that is a monotone control system with respect to the southeast order on both state and all inputs $(w_1^1, \widehat{w}_1^1, \widehat{w}_2^1, w_2^1), \dots, (w_1^n, \widehat{w}_1^n, \widehat{w}_2^n, w_2^n)$.

Observe that $\underline{G}(x, x) = \overline{G}(x, x) = g(x)$. Then, with $W = \widehat{W} = 0$ and initial state $x(0) = x_0$, solutions to $\dot{x} = f'(x, 0)$ are also solutions to $\dot{x} = f(x, g(x))$. Additionally, (Equation 2.42) consists of two copies of this true solution when $x(0) = \widehat{x}(0) = x_0$. Thus, given some $W_1, W_2 \preceq 0$ and $\widehat{W}_1, \widehat{W}_2 \succeq 0$, we can apply the fundamental result of mixed monotone systems theory (Equation 2.6)

for solutions of (Equation 2.42). Namely, that the hyperrectangle defined by the trajectory

$$\Phi^e(t; \underline{x}, \bar{x}, W_1, \widehat{W}_1, W_2, \widehat{W}_2) \quad (2.43)$$

of (Equation 2.42) overapproximates the true reachable set of the system, i.e.

$$\phi(t, x_0) \in \llbracket \Phi^e(t; \underline{x}, \bar{x}, W_1, \widehat{W}_1, W_2, \widehat{W}_2) \rrbracket \quad (2.44)$$

for $x_0 \in [\underline{x}, \bar{x}]$.

Now, consider $(x'(t), \widehat{x}'(t))$ as the solution to (Equation 2.38) for any $(x'(0), \widehat{x}'(0)) = (\underline{x}_0, \bar{x}_0)$.

By the definition of \underline{G} and \overline{G} , there exist some $W_1, W_2 \preceq 0$ and $\widehat{W}_1, \widehat{W}_2 \succeq 0$ such that

$$\begin{aligned} \underline{\gamma}(x, \widehat{x}_{[i:x]}) &= \underline{G}(x, \widehat{x}_{[i:x]}) + w_1^i(t) \\ \overline{\gamma}(x, \widehat{x}_{[i:x]}) &= \overline{G}(x, \widehat{x}_{[i:x]}) + \widehat{w}_1^i(t) \\ \underline{\gamma}(x_{[i:\widehat{x}]}, \widehat{x}) &= \underline{G}(x_{[i:\widehat{x}]}, \widehat{x}) + w_2^i(t) \\ \overline{\gamma}(x_{[i:\widehat{x}]}, \widehat{x}) &= \overline{G}(x_{[i:\widehat{x}]}, \widehat{x}) + \widehat{w}_2^i(t) \end{aligned}$$

for all t . Thus, solutions to (Equation 2.38) are equivalent to solutions of (Equation 2.42) with the associated $W_1, \widehat{W}_1, W_2, \widehat{W}_2$ and therefore overapproximate the true reachable set of the system, i.e.,

$$R(T, X_0) \subseteq \llbracket \Phi^{e'}(T; (\underline{x}_0, \bar{x}_0)) \rrbracket = [x'(T), \widehat{x}'(T)]. \quad (2.45)$$

■

Per the natural inclusion requirement in Assumption 1, modifying the disturbance bound calculation in this way produces tighter bounds, which in turn results in tighter reachable set estimates via Theorem 2. We can similarly apply these modified disturbance bounds toward the verification of forward invariant sets as outlined in Theorem 6.

Theorem 6. *Consider the hypotheses of Theorem 3, and suppose $\underline{x}^*, \bar{x}^* \in \mathbb{R}^n, w_j^{i*}, \widehat{w}_j^{i*} \in \mathbb{R}^p, \underline{x}^* \preceq$*

\bar{x}^* and $\underline{w}_j^{i*} \preceq \bar{w}_j^{i*}$ satisfy, for all $i \in \{1, \dots, n\}$, $j \in \{1, 2\}$,

$$\underline{w}_1^{i*} \preceq \underline{\gamma}(\underline{x}^*, \bar{x}_{[i:\underline{x}^*]}^*) \text{ and } \bar{\gamma}(\underline{x}^*, \bar{x}_{[i:\underline{x}^*]}^*) \preceq \bar{w}_1^{i*} \quad (2.46)$$

$$\underline{w}_2^{i*} \preceq \underline{\gamma}(\underline{x}_{[i:\bar{x}^*]}^*, \bar{x}^*) \text{ and } \bar{\gamma}(\underline{x}_{[i:\bar{x}^*]}^*, \bar{x}^*) \preceq \bar{w}_2^{i*} \quad (2.47)$$

and

$$\delta_i(\underline{x}^*, \underline{w}_1^{i*}, \bar{x}^*, \bar{w}_1^{i*}) \succeq 0, \delta_i(\bar{x}^*, \bar{w}_2^{i*}, \underline{x}^*, \underline{w}_2^{i*}) \preceq 0 \quad (2.48)$$

as alternatives to (Equation 2.20) and (Equation 2.21). Then the conclusion of Theorem 3 remains valid, i.e., $[\underline{x}^*, \bar{x}^*]$ is forward invariant for (Equation 2.2).

The proof is similar to that of Theorem 3 and is omitted.

Thus far, we have considered cases where tightening the confidence bounds on the unknown component of the system results in tighter overapproximations of reachable and forward invariant sets. However, we can approach this problem from a different perspective. Regarding the challenge of safety, we can instead consider our forward-invariant sets to be *safety sets*, and attempt to expand these sets as more observations about the unknown components of the system are made. Traditionally, when approaching the problem of safety set expansion, the entire safety set is considered for observation, with expansion theoretically guaranteed by increasing confidence of the unknown behavior over the entire range [18, 26]. However, Theorem 6 allows us to guide our observation selection; to expand the safety set, we only need to update our knowledge of the unknown component of the dynamics near the edge of the current safety set. We showcase this behavior in section 2.6.

2.5 Gaussian Processes for High Probability Uncertainty Bounds

We now propose an approach to construct functions $\underline{\gamma}$ and $\bar{\gamma}$ satisfying Assumption 1 with probability $1 - \eta$. The crux of the approach is to posit the existence of a specific probability distribution over the function space for each of the unknown functions $\{g_i\}_{i=1}^p$ constituting the unknown part of the dynamics.

Assumption 3. *The unknown functions $\{g_i\}_{i=1}^p$, are independent realizations of a Gaussian Process $\mathcal{GP}(0, k(x, x'))$ with zero mean and kernel $k(\cdot, \cdot)$. In addition, observations of the GP are perturbed by additive i.i.d. Gaussian noise $\mathcal{N}(0, \sigma^2)$ with zero mean and variance σ^2 .*

GPs are a form of non-parametric estimators [9] that are extremely powerful and have gained popularity in a broad range of applications including optimization and control [27, 10, 11, 12, 13, 14]. The kernel $k(\cdot, \cdot)$ is a hyperparameter of the model that controls the correlation of the GP over its domain, which can be viewed as an assumption regarding the smoothness of the unknown function g_i .

Assumption 3 essentially states that the unknown function is drawn from a GP, which allows the approximation of the true unknown functions $\{g_i\}_{i=1}^p$ using surrogate functions that 1) provide lower or upper bounds for the true functions with high probability; and 2) can be refined by acquiring additional observations of the GP. Specifically, given noisy observations $\{y_j\}_{j=1}^t$ of the GPs at corresponding points $\{x_j\}_{j=1}^t$ the surrogate functions of interest to approximate g_i are

$$\forall i \in \{1, \dots, p\} \quad \begin{cases} \bar{g}_i^{(t)}(x) := \mu_t(x) + \sqrt{\beta_t} \sigma_t(x) \\ \underline{g}_i^{(t)}(x) := \mu_t(x) - \sqrt{\beta_t} \sigma_t(x) \end{cases} \quad (2.49)$$

where β_t is to be specified later (see Theorem 7), $\mu_t(\cdot)$ is the posterior mean, and $\sigma_t(\cdot)$ is the posterior variance, computed according to the standard GP updates [9]:

$$\mu_t(x) := k_t(x)^T (K_t + \sigma^2 I)^{-1} y \quad (2.50)$$

$$k_t(x, x') := k(x, x') - k_t(x)^T (K_t + \sigma^2 I)^{-1} k_t(x') \quad (2.51)$$

$$\sigma_t^2(x) := k_t(x, x) \quad (2.52)$$

where $k_t(x) := (k(x_1, x), \dots, k(x_t, x))$ and $K_t = [k_t(x_i, x_j)]$. Intuitively, bounds (Equation 2.49) hold with high probability and can be used to create functions $\underline{\gamma}$ and $\bar{\gamma}$ as follows. For all $i \in$

$\{1, \dots, p\}$ and all $t \geq 1$

$$\forall \underline{x} \preceq \bar{x} \quad \begin{cases} \underline{\gamma}_i^{(t)}(\underline{x}, \bar{x}) := \min_{x \in [\underline{x}, \bar{x}]} \underline{g}_i^{(t)}(x), \\ \bar{\gamma}_i^{(t)}(\underline{x}, \bar{x}) := \max_{x \in [\underline{x}, \bar{x}]} \bar{g}_i^{(t)}(x). \end{cases} \quad (2.53)$$

By definition, the functions in (Equation 2.53) satisfy the natural inclusion property in Assumption 1. However, establishing that (Equation 2.3) holds requires a bit more care because the statement has to hold for all states in a subset of \mathbb{R}^n and for all times t at which updates are made to the GP. We introduce the following mild technical assumptions [27].

Assumption 4. *The states x are confined to a compact subset $\mathcal{D} \subset \mathbb{R}^n$ included in a hypercube of edge size r . In addition, there exist constants $a, b > 0$ such that*

$$\forall i \in \{1, \dots, n\}, \quad \forall j \in \{1, \dots, p\}, \quad \Pr \left(\sup_{x \in \mathcal{D}} \left| \frac{\partial g_j}{\partial x_i} \right| > L \right) \leq a e^{-L^2/b^2}. \quad (2.54)$$

The first part of Assumption 4 is relatively mild since we can assume that the trajectories are confined into a (possibly) large hypercube when computing safe reachable sets. The second part of Assumption 4 is also mild and is satisfied by many kernels of interest [27]. By adapting the proof of [27, Theorem 2], we have the following.

Theorem 7. *Assume that the unknown functions $\{g_i\}_{i=1}^p$ satisfy Assumption 3 and Assumption 4.*

Pick $\eta \in (0, 1)$ and set

$$\beta_t := 2 \log \left(\frac{pt^2 \pi^2}{3\eta} \right) + 2n \log \left(t^2 n b r \sqrt{\log \left(\frac{2pna}{\eta} \right)} \right). \quad (2.55)$$

At every step t of the GP update, define a uniform discretization \mathcal{D}_t of the hypercube containing \mathcal{D} with size τ_t^n where $\tau_t := nt^2 b r \sqrt{\log \left(\frac{2npa}{\eta} \right)}$. For every $x \in \mathcal{D}$, define

$$x^{(t,-)} := \sup \{y \in \mathcal{D}_t \mid y \preceq x\}, \quad (2.56)$$

$$x^{(t,+)} := \inf\{y \in \mathcal{D}_t \mid x \preceq y\}. \quad (2.57)$$

For all $i \in \{1, \dots, p\}$ and all $t \geq 1$ define $\forall \underline{x} \preceq \bar{x}$

$$\underline{\gamma}_i^{(t)}(\underline{x}, \bar{x}) := \min_{x \in [\underline{x}^{(t,-)}, \bar{x}^{(t,+)}] \cap \mathcal{D}_t} \underline{g}_i^{(t)}(x) - \frac{1}{t^2}, \quad (2.58)$$

$$\bar{\gamma}_i^{(t)}(\underline{x}, \bar{x}) := \max_{x \in [\underline{x}^{(t,-)}, \bar{x}^{(t,+)}] \cap \mathcal{D}_t} \bar{g}_i^{(t)}(x) + \frac{1}{t^2}. \quad (2.59)$$

Then, with probability at least $1 - \eta$,

$$\forall \underline{x} \preceq \bar{x} \quad \forall x \in [\underline{x}, \bar{x}] \quad \forall t \geq 1 \quad \forall j \in \{1, \dots, p\}, \quad \underline{\gamma}_i^{(t)}(\underline{x}, \bar{x}) \leq g_i(x) \leq \bar{\gamma}_i^{(t)}(\underline{x}, \bar{x}). \quad (2.60)$$

Sketch of proof. Our proof closely follows the analysis of [28, Appendix I, Section 2], which details the result of [27, Theorem 2]. The idea is to combine the finite discretization \mathcal{D}_t of \mathcal{D} at each step t , for which the GP approximations hold, with Assumption 4 to bound the maximum deviation for points outside the discretization.

With the choice of τ_t and $|\mathcal{D}_t| = \tau_t^n$, the choice of β_t ensures [28, Lemma 5.6] that with probability at least $1 - \eta/2$

$$\forall x \in \mathcal{D}_t \quad \forall i \in \{1, \dots, p\} \quad \forall t \geq 1, \quad \underline{g}_i^{(t)}(x) \leq g_i(x) \leq \bar{g}_i^{(t)}(x). \quad (2.61)$$

Set $\tau := b\sqrt{\log \frac{2npa}{\eta}}$ and note that $\frac{\eta}{2} = npa \exp\left(-\frac{\tau^2}{b^2}\right)$. From Assumption 4, with probability at least $1 - \eta/2$

$$\forall i \in \{1, \dots, n\} \quad \forall j \in \{1, \dots, p\} \quad \sup_{x \in \mathcal{D}} \left| \frac{\partial g_i}{\partial x_j} \right| \leq \tau. \quad (2.62)$$

From (Equation 2.62), we have with probability at least $1 - \eta/2$

$$\forall x, x' \in \mathcal{D} \quad \forall i \in \{1, \dots, n\} \quad \forall j \in \{1, \dots, p\} \quad |g_i(x) - g_i(x')| \leq \tau \|x - x'\|_1. \quad (2.63)$$

When obtaining \mathcal{D}_t as a uniform discretization of size τ_t^n of the hypercube containing \mathcal{D} , we then have

$$\forall x \in \mathcal{D} \quad \|x - x^{(t,-)}\|_1 \leq \frac{nr}{\tau_t} \text{ and } \|x - x^{(t,+)}\|_1 \leq \frac{nr}{\tau_t}.$$

Consequently, with probability at least $1 - \eta/2$ we have

$$\forall x \in \mathcal{D} \quad \forall i \in \{1, \dots, n\} \quad \forall j \in \{1, \dots, p\} \quad \begin{cases} |g_i(x) - g_i(x^{(t,+)})| \leq \tau \frac{nr}{\tau_t} = \frac{1}{t^2}, \\ |g_i(x) - g_i(x^{(t,-)})| \leq \tau \frac{nr}{\tau_t} = \frac{1}{t^2}. \end{cases} \quad (2.64)$$

Combining (Equation 2.61) and (Equation 2.64), with probability at least $1 - \eta$,

$$\forall x \in \mathcal{D} \quad \forall i \in \{1, \dots, n\} \quad \forall j \in \{1, \dots, p\} \quad \begin{cases} g_i(x) \leq \bar{g}_i^{(t)}(x^{(t,+)}) + \frac{1}{t^2}, \\ g_i(x) \geq \underline{g}_i^{(t)}(x^{(t,-)}) - \frac{1}{t^2}, \end{cases} \quad (2.65)$$

Consequently, with probability at least $1 - \eta$, $\forall \underline{x} \preceq \bar{x}, \forall x \in [\underline{x}; \bar{x}], \forall j \in \{1, \dots, p\}$ and $\forall t \geq 1$

$$\gamma_i^{(t)}(\underline{x}, \bar{x}) := \min_{x' \in [\underline{x}^{(t,-)}, \bar{x}^{(t,+)}] \cap \mathcal{D}_t} g_i^{(t)}(x') - \frac{1}{t^2} \quad (2.66)$$

$$\leq \min_{x' \in [\underline{x}^{(t,-)}, \bar{x}^{(t,+)}]} g_i(x') \quad (2.67)$$

$$\leq g_i(x) \quad (2.68)$$

$$\leq \max_{x' \in [\underline{x}^{(t,-)}, \bar{x}^{(t,+)}]} g_i(x') \quad (2.69)$$

$$\leq \max_{x' \in [\underline{x}^{(t,-)}, \bar{x}^{(t,+)}] \cap \mathcal{D}_t} \bar{g}_i^{(t)}(x') + \frac{1}{t^2} \quad (2.70)$$

$$:= \bar{\gamma}_i^{(t)}(\underline{x}, \bar{x}), \quad (2.71)$$

which is the desired result. ■

The bounds (Equation 2.58) and (Equation 2.59) of Theorem 7 can be directly inserted into the

formulations outlined in section 2.4 to calculate reachable and forward invariant sets that hold with probability at least $1 - \eta$.

2.6 Case Studies

We first present a case study of a vehicle encountering hazardous road conditions, illustrating the learning capabilities outlined by Theorem 1 and Theorem 2. We then provide an academic example of the algorithm outlined by Theorem 4 and Proposition 1. Finally, we showcase a case study of a multirotor expanding its known safety set in an unknown wind field by taking advantage of Theorem 6. Another example of a multirotor iteratively exploring a wind field by computing safe reachable sets via Theorem 1 is available in the conference paper [23]. Additionally, a code repository is available at https://github.com/gtfactslab/Cao_OJCSYS2022.

2.6.1 Case Study: Vehicle on an Icy Road

Even though the results above focus on the time-invariant system (Equation 2.2) for notational simplicity, mixed monotonicity, and by extension our results, can naturally accommodate systems subject to known, time-varying, exogenously defined inputs for the reachable set overapproximation, as we demonstrate in this case study.

We consider the kinematic planar bicycle model [29] for abstracting the dynamics of a vehicle. Under nominal conditions, the model relates the positional coordinates X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(u_2)$, and front and rear distances from center of mass l_f and l_r as

$$\begin{aligned}\dot{X} &= v \cos(\psi + \beta(u_2)), \quad \dot{Y} = v \sin(\psi + \beta(u_2)), \\ \dot{\psi} &= \frac{v}{l_r} \sin(\beta(u_2)), \quad \dot{v} = u_1,\end{aligned}\tag{2.72}$$

where

$$\beta(u_2) = \arctan\left(\frac{l_r}{l_f + l_r} \tan(u_2)\right),\tag{2.73}$$

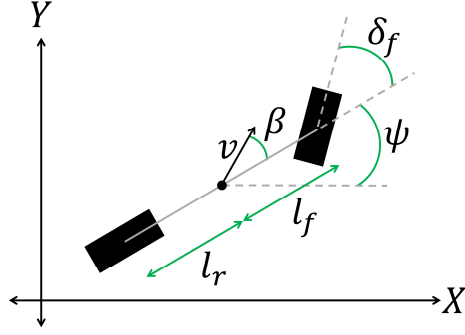


Figure 2.1: The planar bicycle model has positions X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(\delta_f)$, front and rear distances from center of mass l_f and l_r . The inputs to the system are the desired acceleration u_1 and front steering angle $u_2 = \delta_f$.

with inputs to the system being the desired acceleration u_1 and steering angle u_2 . This model is visualized in Figure 2.1.

We consider a scenario where this system is suddenly subject to road conditions that affect the friction between the tires and the road surface, resulting in a disturbance in the velocity dynamics in (Equation 2.72) so that the resulting velocity update dynamics are given by

$$\dot{v}_{\text{actual}} = u_1 + g(u_1), \quad (2.74)$$

where $g(u_1)$ constitutes the unknown effect of the changed friction coefficient between the tires and road.

We assume the inputs $u_1(t)$ and $u_2(t)$ follow the fixed braking and turning maneuver shown in Figure 2.2. We also assume uncertainty in the initial state of the system such that the initial X and Y positions are accurate within 0.5m, and the heading angle is accurate within 0.05rad. The change in the system dynamics (i.e. the road becoming slippery) occurs at time $t = 0$. As shown by the dashed lines in the left plots of Figure 2.3, this disturbance behavior is enough to cause the actual position of the vehicle to be notably different by the end of the 1.6s maneuver.

A decomposition function for the planar bicycle model dynamics with added unknown disturbance (Equation 2.72)-(Equation 2.74) is given as follows. To accommodate the inputs, the associated terms in the decomposition function must simply adhere to requirement 4 in Assumption 2,

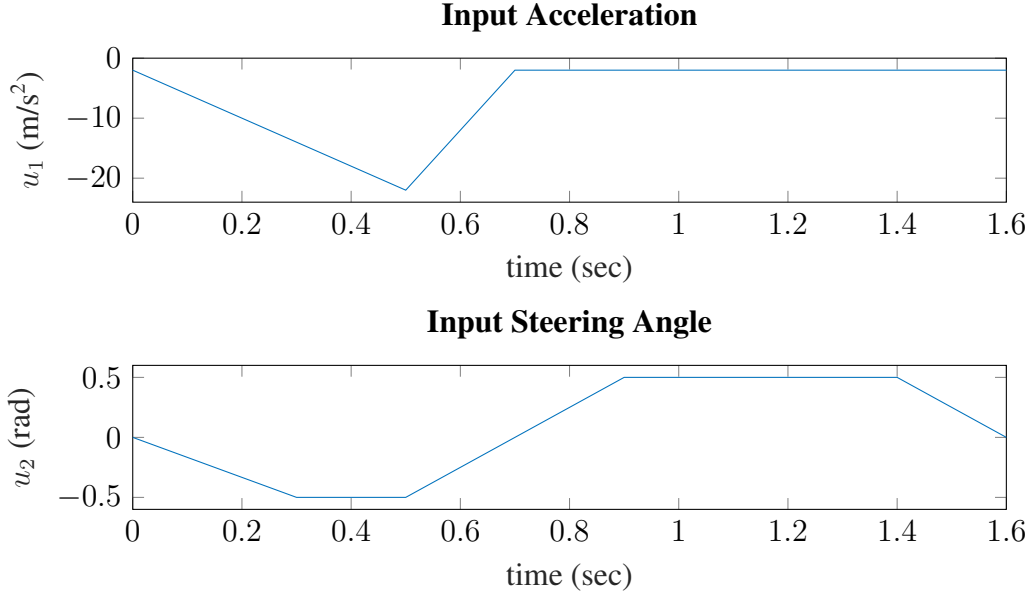


Figure 2.2: Known acceleration and steering angle inputs to the vehicle.

i.e. $\frac{\partial \delta_i}{\partial u_k}(x, u, w, \hat{x}, \hat{u}, \hat{w}) \geq 0$ and $\frac{\partial \delta_i}{\partial \hat{u}_k}(x, u, w, \hat{x}, \hat{u}, \hat{w}) \leq 0$. We take

$$\delta(x, u, w, \hat{x}, \hat{u}, \hat{w}) = \begin{bmatrix} d^X \\ d^Y \\ d^\psi \\ d^v \end{bmatrix} \quad (2.75)$$

where

$$d^X = d^{b_1 b_2} \left(\begin{array}{c} v \\ d^{\cos}(\psi + \beta(\delta_f), \hat{\psi} + \beta(\hat{\delta}_f)) \\ \hat{v} \\ d^{\cos}(\hat{\psi} + \beta(\hat{\delta}_f), \psi + \beta(\delta_f)) \end{array} \right)$$

$$d^Y = d^{b_1 b_2} \left(\begin{array}{c} v \\ d^{\sin}(\psi + \beta(\delta_f), \hat{\psi} + \beta(\hat{\delta}_f)) \end{array} \right)$$

$$d^\psi = d^{b_1 b_2} \left(\begin{array}{c} \left[\begin{array}{c} \hat{v} \\ d^{\sin}(\hat{\psi} + \beta(\hat{\delta}_f), \psi + \beta(\delta_f)) \end{array} \right] \\ \left[\begin{array}{c} v \\ d^{\sin}(\beta(\delta_f), \beta(\hat{\delta}_f)) \end{array} \right], \left[\begin{array}{c} \hat{v} \\ d^{\sin}(\beta(\hat{\delta}_f), \beta(\delta_f)) \end{array} \right] \end{array} \right)$$

$$d^v = u_1 + w,$$

where, for vectors $b, \hat{b} \in \mathbb{R}^2$,

$$d^{b_1 b_2}(b, \hat{b}) = \begin{cases} \min\{b_1 b_2, \hat{b}_1 b_2, b_1 \hat{b}_2, \hat{b}_1 b_2\}, & \text{if } b \preceq \hat{b} \\ \max\{b_1 b_2, \hat{b}_1 b_2, b_1 \hat{b}_2, \hat{b}_1 b_2\}, & \text{if } \hat{b} \preceq b, \end{cases}$$

and d^{\sin}, d^{\cos} are tight decomposition functions for the respective trigonometric functions, which take the forms of equations (Equation 2.76) and (Equation 2.77).

By modeling $g(u)$ as a GP, we attain high-confidence bounds $\underline{\gamma}(u, \hat{u}), \bar{\gamma}(u, \hat{u})$ on the behavior of $g(u)$ which can be inserted into (Equation 2.75). Furthermore, since the input u is known (i.e. $u = \hat{u}$), the computations of $\underline{\gamma}, \bar{\gamma}$ along the system trajectory are simplified. For this example, the $\pm 3\sigma$ bounds of the GP are used. This decomposition function is then used to create the embedding system which provides hyperrectangular overapproximations of the reachable set of the system at the end of the maneuver using Theorem 1. Every 0.1 seconds throughout the maneuver, an observation is collected about the disturbance's behavior, allowing for improved reachable set overapproximation from the updated embedding system.

$$d^{\sin}(x, \hat{x}) := \tag{2.76}$$

$$\left\{ \begin{array}{ll}
\sin(x), & \text{if } (\cos(x), \cos(\hat{x})) \succeq 0 \\
& \text{and } |x - \hat{x}| \leq \pi \\
\sin(\hat{x}), & \text{if } (\cos(x), \cos(\hat{x})) \preceq 0 \\
& \text{and } |x - \hat{x}| \leq \pi \\
\text{sign}(x - \hat{x}), & \text{if } |x - \hat{x}| \geq 2\pi \\
\text{sign}(x - \hat{x}), & \text{if } \cos(x) \leq 0 \leq \cos(\hat{x}) \\
& \text{and } |x - \hat{x}| \leq 2\pi \\
\text{sign}(x - \hat{x}), & \text{if } \cos(x) \cos(\hat{x}) \geq 0 \\
& \text{and } \pi \leq |x - \hat{x}| \leq 2\pi \\
\min\{\sin(x), \sin(\hat{x})\}, & \text{if } x \leq \hat{x} \\
& \text{and } \cos(x) \geq 0 \geq \cos(\hat{x}) \\
& \text{and } |x - \hat{x}| \leq 2\pi \\
\max\{\sin(x), \sin(\hat{x})\}, & \text{if } x \geq \hat{x} \\
& \text{and } \cos(x) \geq 0 \geq \cos(\hat{x}) \\
& \text{and } |x - \hat{x}| \leq 2\pi,
\end{array} \right.$$

$$d^{\cos}(x, \hat{x}) := d^{\sin}\left(x + \frac{\pi}{2}, \hat{x} + \frac{\pi}{2}\right) \tag{2.77}$$

To serve as a baseline, we leverage the Level Set Toolbox [30] and CORA [31] to provide approximations of the reachable set. The state-dependent disturbance behavior can be added in the Level Set Toolbox by including the calculation of the worst-case disturbance bound in the Hamiltonian and partial functions. However, due to the large state space as well as the change in scale from the initial set to the final set, the Level Set Toolbox takes several hours to return a result, and moreover this result was inaccurate, most likely due to numerical issues. Out of the box, CORA does not allow disturbance bounds to evolve in the state-dependent way as described by the

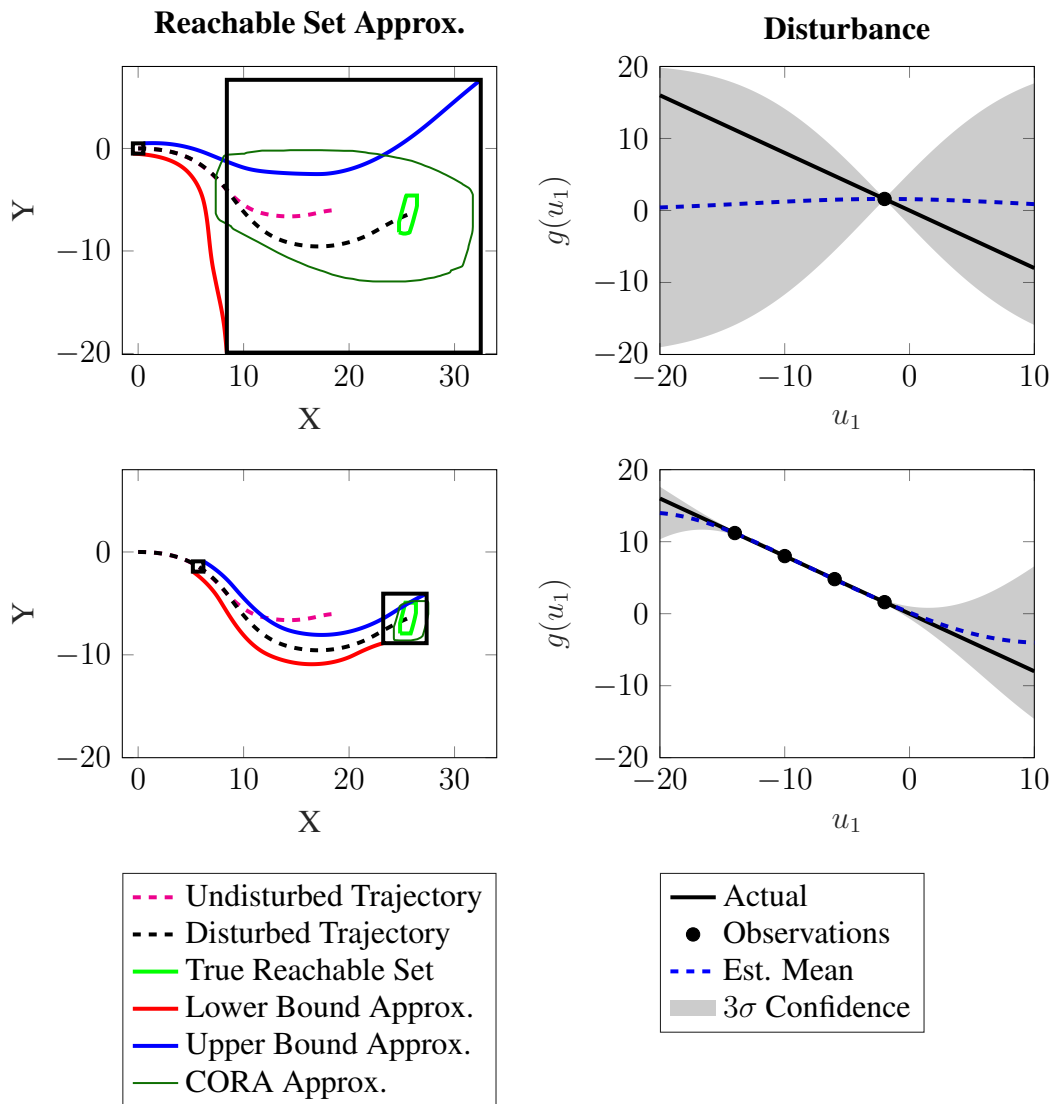


Figure 2.3: Visualization of the hyperrectangular overapproximation of forward reachable set using mixed monotonicity in the autonomous vehicle case study. The disturbance affecting the system is such that the true trajectory of the system notably differs from the undisturbed trajectory (left column, dotted lines). Initially, the system has low knowledge of the disturbance behavior, resulting in a conservative overapproximation of the true reachable set of the vehicle (top row). After obtaining several observations, the reachable set overapproximation is able to tightly approximate the true reachable set (bottom row).

problem, and we are unaware of any other toolboxes that can do so. In CORA, the state-dependent disturbance can be approximated by iteratively solving for the reachable set over a small timestep and inserting the worst-case bounds that will be found during that timestep.

As shown in Figure 2.3, the initial overapproximation of the reachable set is conservative due to having little knowledge about the disturbance behavior. However, after 0.3 seconds, the embedding system is able to provide a much tighter approximation of the reachable set. Additionally, the update of the reachable set overapproximation takes approximately 0.02 seconds on a personal computer with prototype code written in MATLAB, demonstrating the potential for real-time deployment. By contrast, the set approximations given by CORA are less conservative, but take around 600 seconds (10 minutes) to compute on the same machine.

2.6.2 Example: Equilibrium Sequence Convergence

We illustrate Theorem 4 and Proposition 1 via an academic example. Consider the system

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = F(x, w) = \begin{bmatrix} -x_1 - x_1^3 - x_2 - w \\ -x_2 - x_2^3 + x_1 + w^3 \end{bmatrix} \quad (2.78)$$

with state $x = (x_1, x_2) \in \mathbb{R}^2$ and disturbance $w \in \mathbb{R}$. We assume that w is bounded by the parameterized functions $\underline{\gamma}(\underline{x}, \bar{x}; m) \leq w \leq \bar{\gamma}(\underline{x}, \bar{x}; m)$,

$$\underline{\gamma}(\underline{x}, \bar{x}; m) = -m(\max\{|\underline{x}_1|, |\bar{x}_1|\}) \quad (2.79)$$

$$\bar{\gamma}(\underline{x}, \bar{x}; m) = m(\max\{|\underline{x}_1|, |\bar{x}_1|\}) \quad (2.80)$$

where m is a slope parameter.

The system (Equation 2.78) is mixed monotone with respect to the decomposition function

$$\delta(x, w, \hat{x}, \hat{w}) = \begin{bmatrix} -x_1 - x_1^3 - \hat{x}_2 - \hat{w} \\ -x_2 - x_2^3 + x_1 + w^3 \end{bmatrix}. \quad (2.81)$$

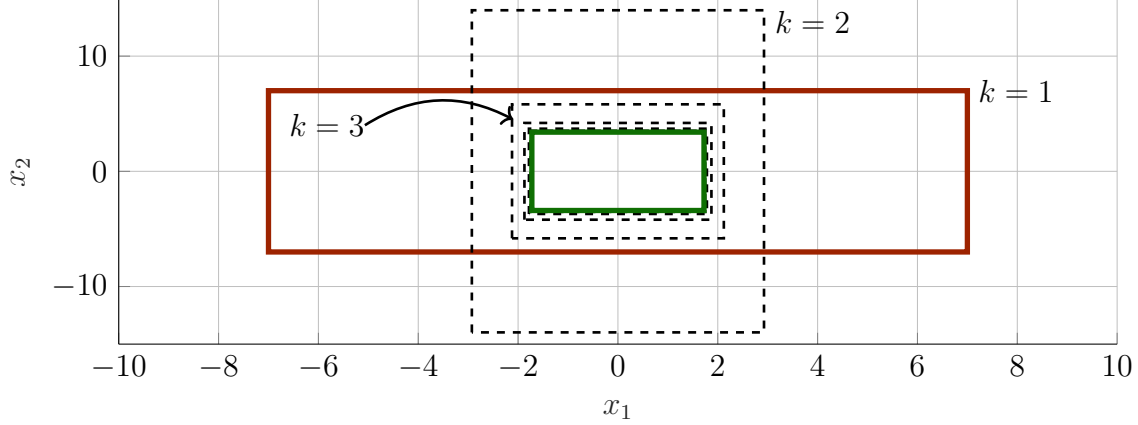


Figure 2.4: Demonstration of Theorem 4 using embedding system (Equation 2.82) and disturbance bounds (Equation 2.79) and (Equation 2.80), $m = 2$. $[\underline{x}^1, \bar{x}^1]$ is initialized to $[(-7, -7), (7, 7)]$, shown in red (outermost solid), and the subsequent steps in the sequence are plotted in black (dashed), with the final converged hyperrectangle shown in green (innermost solid). At $k = 2$, the sequence fulfills (Equation 2.26), from which we conclude that the remaining subsequence is ordered and converges from the second part of Theorem 4. The resulting hyperrectangle, $[(-1.72, -3.40), (1.72, 3.40)]$ is an equilibrium of (Equation 2.82) and is thus forward invariant for (Equation 2.78) per Theorem 4.

Thus, we form the associated embedding system using (Equation 2.7),

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e(x, \hat{x}) := \begin{bmatrix} -x_1 - x_1^3 - \hat{x}_2 - \bar{\gamma}(x, \hat{x}; m) \\ -x_2 - x_2^3 + x_1 + \underline{\gamma}(x, \hat{x}; m)^3 \\ -\hat{x}_1 - \hat{x}_1^3 - x_2 - \underline{\gamma}(x, \hat{x}; m) \\ -\hat{x}_2 - \hat{x}_2^3 + \hat{x}_1 + \bar{\gamma}(x, \hat{x}; m)^3 \end{bmatrix}. \quad (2.82)$$

We then consider two scenarios in which a forward invariant set of (Equation 2.78) is calculated by determining an equilibrium of the system. In the first scenario we take $m = 2$, then in the second scenario we initialize the equilibrium sequence to the results of the first scenario and take $m = 1$. These scenarios are demonstrated in Figure 2.4 and Figure 2.5, respectively.

2.6.3 Case Study: Multirotor Exploring a Wind Field

We consider a planar multirotor model, illustrated in Figure 2.6, for a multirotor aerial vehicle that is constrained to move in a vertical plane. The horizontal and vertical position of the multirotor are

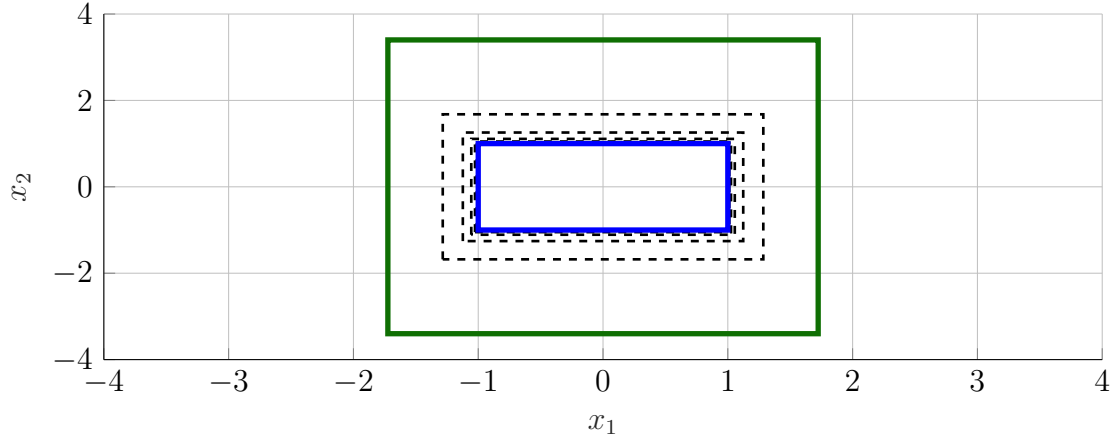


Figure 2.5: Demonstration of Proposition 1 using embedding system (Equation 2.82) and disturbance bounds (Equation 2.79) and (Equation 2.80), $m = 1$. Note that $(\underline{\gamma}, \bar{\gamma}) = (\underline{\gamma}(x, \bar{x}; 1), \bar{\gamma}(x, \bar{x}; 1))$ and $(\underline{\gamma}', \bar{\gamma}') = (\underline{\gamma}(x, \bar{x}; 2), \bar{\gamma}(x, \bar{x}; 2))$ fulfill (Equation 2.33). $[x^1, \bar{x}^1]$ is initialized to the result from the first scenario, $[(-1.72, -3.40), (1.72, 3.40)]$, shown in green (outermost solid), and the subsequent steps in the sequence are plotted in black (dashed), with the final converged hyperrectangle shown in blue (innermost solid). The resulting hyperrectangle, $[(-1.00, -1.00), (1.00, 1.00)]$ is an equilibrium of (Equation 2.82) and is thus forward invariant for (Equation 2.78) per Theorem 4.

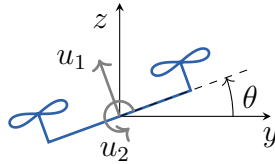


Figure 2.6: The planar multicopter model has horizontal position y , vertical position z , and roll angle θ . The inputs are thrust u_1 in the direction perpendicular to the line segment connecting the rotors and roll angle velocity u_2 . The five dimensional state x consists of y, z, θ , and the derivatives $v_y = \dot{y}, v_z = \dot{z}$, so that $x = [y \ v_y \ z \ v_z \ \theta]^T$.

denoted y and z , and the roll angle is denoted θ . The system has two inputs, thrust u_1 acting at the center of mass in the direction $\begin{bmatrix} -\sin \theta & \cos \theta \end{bmatrix}^T$, and the roll angular velocity u_2 . We also assume that the system is subject to input saturation, gravitational acceleration a_g , and an unknown force due to wind. This wind force affects the acceleration in both the horizontal and vertical directions, and the force in each direction is a function of its associated state (i.e. the horizontal wind force is a function of position y and the vertical wind force is a function of altitude z). The resulting dynamics with normalized mass and moment of inertia are given by

$$\begin{aligned}\ddot{y} &= -u_1 \sin \theta + g_1(y) \\ \ddot{z} &= u_1 \cos \theta - a_g + g_2(z) \\ \dot{\theta} &= u_2\end{aligned}\tag{2.83}$$

where g_1 and g_2 constitute the unknown wind forces in the horizontal and vertical directions, respectively. We estimate g_1 and g_2 using GPs with a radial basis function kernel, and obtain high confidence bounds by considering posterior estimates up to two standard deviations from the mean.

To control the system, we employ feedback linearization to transform the system into a set of triple integrators in the y and z directions. We then perform an eigenvector transformation and utilize the method described in [1] to derive a decomposition function for the resulting feedback linearized system. This decomposition function allows us to determine high probability forward invariant sets that account for the potential disturbance modeled by the GPs. We then develop an operational controller and a safety controller using the feedback linearized system. The operational controller utilizes proportional-integral-derivative (PID) control to drive the system to a target point, while the safety controller uses proportional-derivative (PD) control to drive the system to the origin. During operation, an invariant safe set is computed using Theorem 5 for the closed loop system with safety controller, such that the safety controller does not produce inputs that will saturate. The overall control strategy is to execute the operational controller action to explore the safety set and collect measurements, and to switch to the safety controller if the sys-

tem enters within some distance of the safety set boundary in order to maintain safety with high probability.

We assume there exist obstacles which the system must avoid as shown in Figure 2.7. Each GP is initialized with observations near the starting point of the multirotor vehicle. Using Theorem 5 and numerically searching for states in the embedding system that satisfy Theorem 6, we find the largest invariant set that does not intersect the obstacles and thus can be used as a safety set, as well as the smallest invariant set that intersects the obstacles and therefore cannot be used as a safety set, as shown in Figure 2.7. We also leverage Theorem 1 and Theorem 3 and the same numerical search method to find another invariant set, which can be seen in the figure to be smaller and thus more restrictive.

We then task the system with safely collecting observations about the wind's behavior in order to update the GPs, allowing for iterative computation of new, larger safety sets. Recalling that Theorem 6 allows us to focus observation points near the edge of the safety set, we have the system fly to observation points near the corners of the current safety set (marked by the asterisks in the figures), take measurements of the disturbance, and then update the GPs and calculate a new forward invariant set. This process continues iteratively, allowing for the multirotor to safely explore larger areas of the state space with high probability guarantees of avoiding unsafe regions, as shown in Figure 2.7. For comparison, we use the same procedure with Theorem 1 and Theorem 3. After the same number of updates, we can again see in the figure that this produces a more restrictive invariant set. On average, the sets found by Theorem 6 took around 1.8 seconds to produce on a personal computer with code written in MATLAB, while the sets found by Theorem 3 took around 0.8 seconds, showcasing the tradeoff between these theorems.

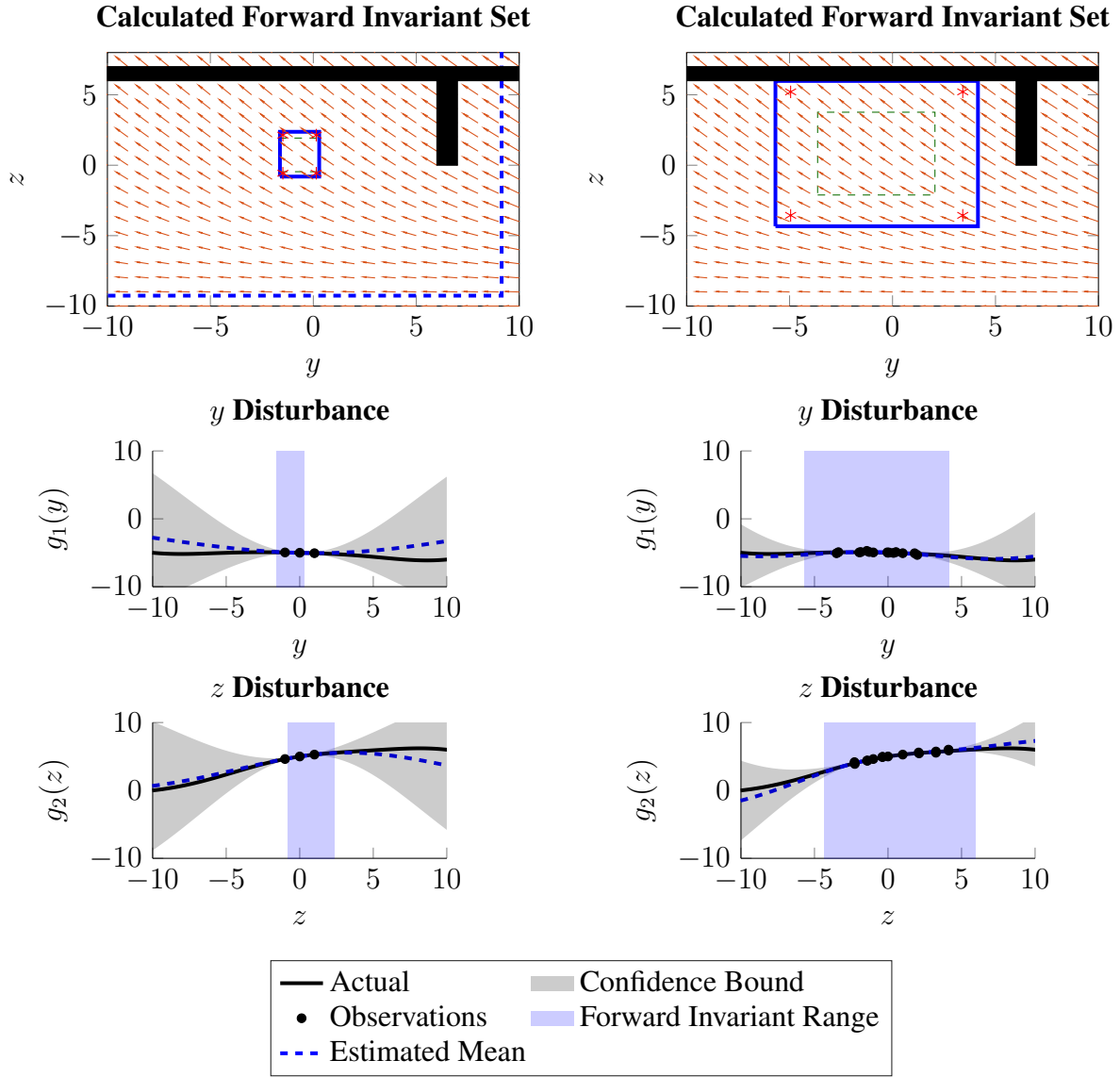


Figure 2.7: The planar multirotor operates in an unknown wind field with several obstacles and is initialized with some observations (2nd row) of the wind’s behavior (1st row, arrows). Using Theorem 6, we identify two invariant sets for the closed-loop safety controller (1st row, blue solid and dashed rectangles). However, only the smaller (blue solid) set can be used initially as a safety set because the larger set (blue dashed) intersects obstacles. Additionally, the largest acceptable set (1st row, green dashed) produced by Theorem 3 is more restrictive. After collecting several rounds of observations (4th row), the multirotor can now safely explore near the obstacles without collision, as denoted by the new safety set (3rd row, solid blue). Again, note that set produced by Theorem 3 (3rd row, green dashed) is more restrictive.

CHAPTER 3

SAFE LEARNING-BASED PREDICTIVE CONTROL FROM EFFICIENT REACHABILITY

3.1 Introduction

When the dynamics of a controlled system are not fully known, a common approach is to apply control actions to explore and observe the behavior of the system and adjust the control strategy as new information is collected. However, for systems with safety constraints that restrict allowable regions of the state space, the process of collecting observations must be designed so as not to lead to unsafe behavior.

Learning and exploration with safety guarantees has been considered in [16], which proposes a discrete-time MPC algorithm that is guaranteed to be safe with high probability by ensuring that a path back to safety exists at every timestep. Alternatively, [32] presents a learning algorithm that explicitly considers safety defined in terms of Lyapunov stability guarantees, [19] proposes a general safety framework based on Hamilton-Jacobi reachability methods, [33, 34, 35] synthesize control barrier functions online to guarantee safety, and [36] achieves safety by estimating the Lipschitz constant of the disturbance. Other works, such as [37, 38, 39], explore learning and updating safety sets in an online manner. For MPC-based approaches, proposed frameworks are robust to certain types of uncertainty, for example by assuming a known Lipschitz constant [40], assuming the uncertainty is parametric [41], or applying MPC to iterative learning control [42].

We draw from the problem setup proposed in [16] and consider a nonlinear dynamical system whose dynamics are not fully known. As in [16], we estimate the unknown component using GP regression. Exploration of the state space is allowed so long as a feasible return trajectory is available that returns the system to a known safe set.

In this chapter, we continue to consider systems in continuous-time subject to state-dependent

unknown components that enter the dynamics nonlinearly. We leverage the mixed monotonicity property of dynamical systems and utilize the results from the previous chapter (including [23, 22]) to obtain hyperrectangular overapproximations of the reachable sets of the system that hold with high probability. We recall that these overapproximations are obtained by computing a single trajectory of an appropriately constructed *embedding system* that is an ordinary differential equation with twice the dimension of the original system.

Comparing to existing approaches, we consider continuous-time systems with nonlinear disturbances and we use reachability techniques that are computationally efficient and scalable, as demonstrated on a multicopter case study with six states in [43]. Moreover, this approach avoids excessive conservatism that often occurs when linearizing the dynamics and outerbounding the linearization error using the Lipschitz constant of the dynamics [21]. Lastly, we explicitly consider the goal of reaching a target region of the state-space while avoiding an unsafe region. We pose our algorithm for safe exploration and goal reaching as a continuous-time model predictive control problem.

The rest of this chapter is structured as follows: section 3.2 formally defines the problem, after which we develop a controller that solves this problem in section 3.3. We demonstrate its efficacy on several case studies in section 3.4: an autonomous vehicle traveling on an icy road, a planar multicopter operating in a wind field, and a motorboat navigating a river.

3.2 Problem Setup

Consider the continuous-time nonlinear dynamical system

$$\dot{x} = f(x, u, w) \tag{3.1}$$

with f differentiable where $x \in \mathbb{R}^n$ is the system state, $u \in \mathcal{U} \subset \mathbb{R}^m$ is the input constrained to take values in \mathcal{U} , and $w \in \mathbb{R}^p$ is an unknown, state-dependent component of the dynamics so that $w_i = g_i(x)$ where g_i is unknown. Throughout, we assume the input constraint set has the form

$\mathcal{U} = [\underline{u}, \bar{u}]$ for some $\underline{u}, \bar{u} \in \mathbb{R}^m$, $\underline{u} \preceq \bar{u}$, that is, \mathcal{U} is a hyperrectangle defined by corners \underline{u} and \bar{u} .

We denote by $\phi(t, x_0, \pi)$ the resulting closed-loop state trajectory of (Equation 3.1) under control strategy $u = \pi(t, x)$ when $w = g(x)$ and the system is initialized at x_0 at time 0. If π is time-invariant, we write $\pi(x)$ instead. For this chapter only, we also consider the following assumption.

Assumption 5. *There exists a known subset of the state space $\mathcal{X}_{\text{unsafe}} \subset \mathbb{R}^n$ which must be avoided. There also exists a known safe set $\mathcal{X}_{\text{safe}} \subset \mathbb{R}^n$ and corresponding time-invariant safety controller π_{safe} with $\pi_{\text{safe}}(x) \in \mathcal{U}$ for all $x \in \mathbb{R}^n$ such that, if the system is initialized in $\mathcal{X}_{\text{safe}}$, it avoids $\mathcal{X}_{\text{unsafe}}$, i.e.*

$$\phi(t, x_0, \pi_{\text{safe}}) \in (\mathcal{X}_{\text{unsafe}})^c \quad \forall t \geq 0, \quad \forall x_0 \in \mathcal{X}_{\text{safe}}. \quad (3.2)$$

Our objective is to control the system to a goal region while avoiding the unsafe region.

Problem 3. *Consider a system as in (Equation 3.1) with specified initial condition $x_0 \in \mathcal{X}_{\text{safe}}$ and input constraints \mathcal{U} . Given a goal region $\mathcal{X}_{\text{goal}} \subset \mathbb{R}^n$, compute a feedback control strategy $u = \pi(t, x)$ that reaches the goal while avoiding the unsafe region $\mathcal{X}_{\text{unsafe}}$, i.e.,*

$$\forall t \geq 0, \phi(t, x_0, \pi) \in (\mathcal{X}_{\text{unsafe}})^c \quad (3.3)$$

$$\exists T > 0 \text{ s.t. } \phi(T, x_0, \pi) \in \mathcal{X}_{\text{goal}}. \quad (3.4)$$

In general, we are interested in scenarios in which $\mathcal{X}_{\text{goal}}$ does not intersect $\mathcal{X}_{\text{safe}}$ so that we cannot achieve our objective by remaining within $\mathcal{X}_{\text{safe}}$. Thus, while the safety controller π_{safe} achieves (Equation 3.3), it generally will not achieve (Equation 3.4).

Our proposed control approach is to incrementally move towards the goal while ensuring the system is always able to safely return to $\mathcal{X}_{\text{safe}}$ if needed, until it can be guaranteed that the system can safely reach the goal. This safe return and guaranteed reach to the goal is ensured via a nonlinear MPC scheme which directly optimizes for a control input in both cases and incorporates uncertainty from the unknown component $g(x)$ of the dynamics. While moving towards the

goal, the system is able to collect information about its dynamics and reduce the uncertainty in its estimate of $g(x)$, allowing it more freedom to safely explore. We next formalize this approach.

3.3 Safe Learning Algorithm

In this section we develop a safe control scheme that is safe with high probability by leveraging the mixed monotonicity property of dynamical systems to calculate high-probability reachable sets, then utilize an MPC formulation to solve for a control strategy that always has a path back to safety. We define the reachable set of (Equation 3.1) at time $t = T$ initialized from any state $x_0 \in X_0$ with control policy u as

$$R(T, X_0, u) = \{\phi(T, x_0, u) \mid x_0 \in X_0\}. \quad (3.5)$$

3.3.1 Mixed Monotonicity

We preserve Assumption 2 and adjust the requirements on the decomposition function δ to apply mixed monotonicity on the system (Equation 3.1). The new requirements are as follows:

1. For all x and all w , $\delta(x, u, w, x, w) = f(x, u, w)$;
2. For all $i, j \in \{1, \dots, n\}, i \neq j$, $\frac{\partial \delta_i}{\partial x_j}(x, u, w, \hat{x}, \hat{w}) \geq 0$ for all $x, \hat{x}, u, w, \hat{w}$;
3. For all $i, j \in \{1, \dots, n\}$, $\frac{\partial \delta_i}{\partial \hat{x}_j}(x, u, w, \hat{x}, \hat{w}) \leq 0$ for all $x, \hat{x}, u, w, \hat{w}$;
4. For all $i \in \{1, \dots, n\}$ and all $k \in \{1, \dots, p\}$, $\frac{\partial \delta_i}{\partial w_k}(x, u, w, \hat{x}, \hat{w}) \geq 0$ & $\frac{\partial \delta_i}{\partial \hat{w}_k}(x, u, w, \hat{x}, \hat{w}) \leq 0$ for all $x, \hat{x}, u, w, \hat{w}$.

We form our embedding system accordingly, and add our previously derived high-probability bounds to form

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e(x, u, \hat{x}) := \begin{bmatrix} \delta(x, u, \underline{\gamma}(x, \hat{x}), \hat{x}, \bar{\gamma}(x, \hat{x})) \\ \delta(\hat{x}, u, \bar{\gamma}(x, \hat{x}), x, \underline{\gamma}(x, \hat{x})) \end{bmatrix}. \quad (3.6)$$

3.3.2 Safe With High Probability MPC

We sample the embedding system (Equation 3.6) with timestep h such that at each step $k = t/h$,

$$\begin{bmatrix} x[k+1] \\ \hat{x}[k+1] \end{bmatrix} = \Phi^e(h; (x[k], \hat{x}[k]), \pi_k) \quad (3.7)$$

where π_k is the controller applied from time kh to $(k+1)h$. Below, we assume π_k is a constant policy $\pi_k(t, x) \equiv u_k$ for some $u_k \in \mathcal{U}$ to be designed by an MPC scheme. Thus, taking $\hat{R}_k = [x[k], \hat{x}[k]]$ overapproximates the reachable set of (Equation 3.1) at time $t = kh$ with high probability (i.e. with probability at least $1 - \eta$).

We then use these overapproximations and formulate the following MPC scheme which satisfies the safety condition (Equation 3.3):

$$\underset{\Pi = \{u_0, \dots, u_D\}}{\text{minimize}} \quad J_{k, \text{obj}}(\hat{R}_0, \dots, \hat{R}_D) \quad (3.8)$$

subject to:

$$\begin{aligned} & \text{(Equation 3.7), } x[0] = \hat{x}[0] \text{ given, } u_d \in \mathcal{U} && \forall d \in \{0, \dots, D-1\} \\ & \hat{R}_d = [x[d], \hat{x}[d]], \hat{R}_D \subset \mathcal{X}_{\text{obj}} && \forall d \in \{0, \dots, D\} \\ & [x(t), \hat{x}(t)] \subset (\mathcal{X}_{\text{unsafe}})^C && \forall t \in [0, T] \end{aligned}$$

where $\text{obj} \in \{\text{goal}, \text{safe}\}$. The control strategy incorporating the above MPC scheme is outlined in algorithm 1. This strategy optimizes for desired behavior based on the cost functions $J_{k, \text{goal}}$ and $J_{k, \text{safe}}$ which are designed to prioritize goal-reaching and exploration, respectively. If (Equation 3.8) is feasible when $\text{obj} = \text{goal}$, then Π contains a control input for each timestep that altogether are guaranteed with high probability to drive the system into $\mathcal{X}_{\text{goal}}$ while avoiding $\mathcal{X}_{\text{unsafe}}$. Thus, the entire control strategy Π is executed immediately and the algorithm terminates.

Otherwise, the algorithm attempts to solve (Equation 3.8) with $\text{obj} = \text{safe}$. If this MPC problem is feasible, Π contains a set of control inputs that explores the state space while guaranteeing

with high probability that the system will avoid $\mathcal{X}_{\text{unsafe}}$ and return to $\mathcal{X}_{\text{safe}}$. Thus, the algorithm saves the entire strategy as Π_k . If the problem is not feasible, the algorithm copies the unexecuted actions from the previous saved strategy Π_{k-1} and appends the safety action π_{safe} . The previously saved strategy Π_{k-1} must either end in $\mathcal{X}_{\text{safe}}$ or be the result of applying π_{safe} for all time after starting in $\mathcal{X}_{\text{safe}}$, thus Π_k is guaranteed to be safe with high probability. The algorithm then executes the first action saved in Π_k , and restarts at trying to solve (Equation 3.8) with $\text{obj} = \text{goal}$.

The system may be initially unable to reach the goal, as high uncertainty on the bounds of the unknown behavior may prevent the final reachable set \widehat{R}_K from being contained in $\mathcal{X}_{\text{goal}}$. However, as observations are collected and the bounds $\underline{\gamma}, \overline{\gamma}$ tighten, the reachable set overapproximations also tighten, allowing for finer control over the system and thus allowing for exploration further outside of $\mathcal{X}_{\text{safe}}$ or enabling the system to reach $\mathcal{X}_{\text{goal}}$. We note that, in practice, to ensure the safety condition $[x(t), \widehat{x}(t)] \subset (\mathcal{X}_{\text{unsafe}})^C$ for all $t \in [0, T]$ in (Equation 3.8), we check this condition at a large number of time instances between the sampling times.

Algorithm 1 Resulting Control Scheme

Data: Safety controller π_{safe} , embedding system (Equation 3.6) sampled as (Equation 3.7), bounding functions $\underline{\gamma}, \overline{\gamma}$

$\Pi_0 \leftarrow \{\pi_{\text{safe}}, \dots, \pi_{\text{safe}}\};$

for $k = 0, 1, \dots$ **do**

$(\text{feasible}, \Pi) \leftarrow$ solve MPC problem, $\text{obj} = \text{goal}$;

if *feasible* **then**

 apply $u = \Pi$ to system;

break;

$(\text{feasible}, \Pi) \leftarrow$ solve MPC problem, $\text{obj} = \text{safe}$;

if *feasible* **then**

$\Pi_k \leftarrow \Pi$;

else

$\Pi_k \leftarrow \{\Pi_{k-1,1:D-1}, \pi_{\text{safe}}\};$

$x_{k+1} \leftarrow$ apply $u(t) = \Pi_{k,0}(x(t))$ to (Equation 3.1) until $t = (k + 1)h$;

We then have the following guarantee of safety for all timesteps, even if the MPC problems are infeasible.

Theorem 8. *Given a system (Equation 3.1) under the assumptions made in Problem 3 with $x_0 \in \mathcal{X}_{\text{safe}}$, the control strategy resulting from algorithm 1 is safe with probability at least $1 - \eta$.*

The proof of this theorem is built on the following Lemmas.

Lemma 2. *Given a system (Equation 3.1) under the assumptions made in Problem 3 at timestep $k = 0$ with $x_0 \in \mathcal{X}_{\text{safe}}$, the control strategy resulting from algorithm 1 is safe with probability at least $1 - \eta$.*

Proof. Given the constraints of the MPC problem, the produced reachable sets $[x(t), \hat{x}(t)]$ do not intersect any portion of $\mathcal{X}_{\text{unsafe}}$. As these reachable sets hold with probability at least $1 - \eta$ per Theorem 7, it follows that the overall strategy Π that produces these reachable sets is safe with probability at least $1 - \eta$.

Thus, if the MPC problem (Equation 3.8) is feasible for either $\text{obj} = \text{safe}$ or $\text{obj} = \text{goal}$, the resulting strategy is safe with probability at least $1 - \eta$.

If the MPC problem is infeasible in both cases, algorithm 1 produces a control strategy that only consists of applying π_{safe} . As the system is initialized in $\mathcal{X}_{\text{safe}}$, this guarantees the system's safety per the problem setup. ■

Lemma 3. *Given a system (Equation 3.1) under the assumptions made in Problem 3, and a control strategy computed by algorithm 1 in the previous timestep $k - 1$ which is safe with probability at least $1 - \eta$, the control strategy resulting from algorithm 1 at timestep k is safe with probability at least $1 - \eta$.*

Proof. Again, if the MPC problem (Equation 3.8) is feasible for either $\text{obj} = \text{safe}$ or $\text{obj} = \text{goal}$, the resulting strategy is safe with probability at least $1 - \eta$.

If the MPC problem is infeasible in both cases, algorithm 1 produces a control strategy that appends π_{safe} to the strategy computed in the previous timestep.

We then consider the potential origins of the previous strategy. If the previous strategy was originally computed by solving (Equation 3.8) with $\text{obj} = \text{safe}$, by the constraints of the MPC problem, applying this strategy results in the system avoiding $\mathcal{X}_{\text{unsafe}}$ and ending in $\mathcal{X}_{\text{safe}}$ with probability at least $1 - \eta$. Thus, appending π_{safe} to the end of this strategy preserves this safety via the problem

setup. It is not possible for the previous strategy to have resulted from solving (Equation 3.8) with $\text{obj} = \text{goal}$, as algorithm 1 immediately executes the entire strategy if that problem is feasible.

If the previous strategy is a result of the MPC problem being infeasible in both cases, then that previous strategy will have, at some point, entered $\mathcal{X}_{\text{safe}}$ and then started only executing π_{safe} . Thus, appending π_{safe} to this strategy preserves safety via the problem setup. ■

With the previous Lemmas now proven, we now formally prove Theorem 8.

Proof of Theorem 8. At timestep $k = 0$, the control strategy produced by algorithm 1 is safe with probability at least $1 - \eta$ via Lemma 2, and must be safe with probability at least $1 - \eta$ for every timestep afterward via Lemma 3. Thus, the control strategy produced by algorithm 1 is safe for all timesteps with probability at least $1 - \eta$. ■

Thus, this control scheme solves Problem 3. We emphasize that the key features of algorithm 1 that make it computationally tractable and always safe are: 1) the sequential solving of a goal-reaching strategy and, failing that, a safe strategy; 2) the persistent safety property that there always exists a known safe control action from the previous step that can be executed next if needed; 3) the efficient computation of high probability reachable sets using mixed monotone systems theory.

3.4 Case Studies

In this section we apply algorithm 1 to case studies of a four-dimensional autonomous vehicle operating on an icy road, a six-dimensional planar multirotor operating in a wind field, and a four-dimensional motorboat operating on a river. We task each with safely reaching a goal area in the presence of unknown disturbances (i.e. the ice, wind, and river flow) while avoiding unsafe sets. The case studies were implemented using the Model Predictive Control Toolbox in MATLAB, and a code repository is available at https://github.com/gtfactslab/Cao_ACC2023 for the Autonomous Vehicle and Planar Multirotor case studies, and at https://github.com/gtfactslab/Cao_OptimisticControl for the Boat case study.

3.4.1 Autonomous Vehicle on Icy Road

We consider an autonomous vehicle modeled by the four-dimensional kinematic planar bicycle, which has state $x = [X, Y, \psi, v]^T$ and relates positional coordinates X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(u_2)$, and front and rear distances from center of mass $l_f = 2.2\text{m}$ and $l_r = 3.3\text{m}$ as

$$\begin{aligned}\dot{X} &= v \cos(\psi + \beta(u_2)), & \dot{Y} &= v \sin(\psi + \beta(u_2)), \\ \dot{\psi} &= \frac{v}{l_r} \sin(\beta(u_2)), & \dot{v} &= u_1,\end{aligned}\tag{3.9}$$

where

$$\beta(u_2) = \arctan\left(\frac{l_r}{l_f + l_r} \tan(u_2)\right),\tag{3.10}$$

with inputs being the desired acceleration u_1 and steering angle u_2 . We assume the system is subject to constraints $\mathcal{U} = [(-600, -\pi/3), (600, \pi/3)]$.

The vehicle is operating on a road which varies in friction due to the presence of ice patches. As a result, the actual velocity update dynamics are given by

$$\dot{v}_{\text{actual}} = (1 - g(X, Y))u_1,\tag{3.11}$$

where the true behavior of $g(X, Y) \in [0, 1)$ represents the state-dependent change in friction based on the road surface at that position. We estimate g using a GP with a radial basis kernel, initialized with several points around the starting point of the vehicle, and obtain high-confidence bounds by considering posterior estimates up to three standard deviations from the mean. We set $h = 0.05\text{s}$ and collect an observation of $g(X, Y)$ at every timestep.

The associated decomposition function takes the form

$$\delta(x, u, w, \hat{x}, \hat{w}) = \begin{bmatrix} d^X & d^Y & d^\psi & d^v \end{bmatrix}^T\tag{3.12}$$

$$d^v = d^{b_1 b_2} \left(\begin{bmatrix} 1 - \widehat{w} \\ u_1 \end{bmatrix}, \begin{bmatrix} 1 - w \\ u_1 \end{bmatrix} \right)$$

where, for $b, \widehat{b} \in \mathbb{R}^2$,

$$d^{b_1 b_2}(b, \widehat{b}) = \begin{cases} \min\{b_1 b_2, \widehat{b}_1 b_2, b_1 \widehat{b}_2, \widehat{b}_1 \widehat{b}_2\}, & \text{if } b \preceq \widehat{b} \\ \max\{b_1 b_2, \widehat{b}_1 b_2, b_1 \widehat{b}_2, \widehat{b}_1 \widehat{b}_2\}, & \text{if } \widehat{b} \preceq b, \end{cases}$$

and d^X, d^Y, d^ψ take the same forms as in [22, Section VI-A].

We define the safety set as

$$\mathcal{X}_{\text{safe}} = [(-2, -5, -2\pi, -50), (0, 5, 2\pi, 50)] \quad (3.13)$$

and task the system with entering the goal set

$$\mathcal{X}_{\text{goal}} = [(5, -5, -2\pi, -15), (7, 5, 2\pi, 15)] \quad (3.14)$$

while avoiding $\mathcal{X}_{\text{unsafe}}$, which is the union of a set of hyperrectangles (see the dashed red boxes in Figure 3.1).

We then leverage the following observation:

Observation 1. *Given vectors (x, x') and (y, y') such that $x \preceq x'$ and $y \preceq y'$,*

$$(x, x') \preceq_{SE} (y, y) \iff [x, x'] \cap [y, y'] \neq \emptyset. \quad (3.15)$$

Thus, our safety constraint is equivalently defined as

$$(x(t), \widehat{x}(t)) \not\preceq_{SE} (\bar{x}_{ui}, \underline{x}_{ui}) \quad \forall t \geq 0, i \quad (3.16)$$

where $[\underline{x}_{ui}, \bar{x}_{ui}]$ are the unsafe hyperrectangles. We check that six intermediate reachable sets

between each timestep satisfy $[x(t), \hat{x}(t)] \subset (\mathcal{X}_{\text{unsafe}})^C$ and solve the MPC problem (Equation 3.8) for $D = 4$ timesteps.

We take

$$J_{k,\text{goal}}(\hat{R}_0, \dots, \hat{R}_D) = \sum_{d=0}^D \|C_d - C_{\text{goal}}\| \quad (3.17)$$

where C_d, C_{goal} are the center points of the respective reachable sets and goal. If a feasible solution is found, the set of control actions is executed immediately, as these represent a control strategy that is guaranteed to end in $\mathcal{X}_{\text{goal}}$ with high probability. If this is infeasible, (Equation 3.8) is solved with cost

$$J_{k,\text{safe}}(\hat{R}_0, \dots, \hat{R}_D) = - \sum_{d=0}^D \left(\alpha \sigma(C_d) - \|C_d - C_{\text{goal}}\| e^{-\alpha \sigma(C_d)} \right) \quad (3.18)$$

where $\alpha > 0$ is a user-specified constant. This reverts the objective back to pure exploration, with a bias toward observation points that take the system closer to the goal. Multiplying the bias term $\|C_d - C_{\text{goal}}\|$ by $e^{-\alpha \sigma(C_d)}$ allows the bias to be overcome if the expected information gain is high enough, and the exponential function is specifically chosen to mirror the structure of the GP radial basis kernel.

As shown in Figure 3.1, initially, the system is unable to find a control strategy that is guaranteed to reach the goal while avoiding the unsafe areas with high probability, so it reverts to exploring the state space for the first few timesteps (first and second plots). After sufficient exploration, the controller is able to compute a strategy that reaches the goal, and executes this strategy immediately (third and fourth plot). Thus, at all times, the controller has a safe path to either $\mathcal{X}_{\text{safe}}$ or $\mathcal{X}_{\text{goal}}$ that holds with high probability. On average, it takes approximately 18 minutes to compute the next control action single-threaded on a personal computer, though timing analysis shows that around 60 percent of this computation is spent sampling the GP; these processes are parallelizable, though outside the scope of this work.

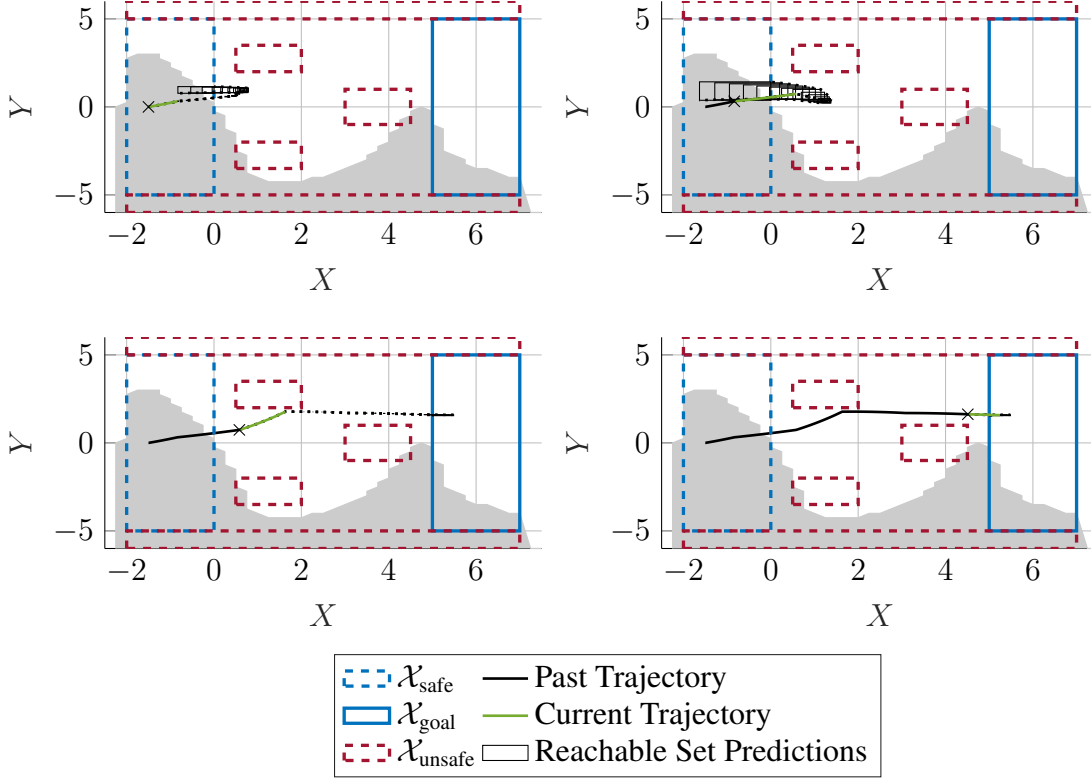


Figure 3.1: Execution of the Autonomous Vehicle case study from [43]. The shaded region is the portion of the road unaffected by ice. The control strategy given by algorithm 1 ensures that the vehicle always has a path back to the safety set until it can find a path to the goal.

3.4.2 Planar Multirotor in Wind Field

We consider a multirotor aerial vehicle constrained to move in a vertical plane. The six-dimensional state x of the planar multirotor system consists of horizontal position y , vertical position z , roll angle θ , and their derivatives, $v_y = \dot{y}$, $v_z = \dot{z}$, $\omega = \dot{\theta}$, so that $x = \begin{bmatrix} y & v_y & z & v_z & \theta & \omega \end{bmatrix}^T$. The two inputs are thrust u_1 acting at the center of mass in the direction $\begin{bmatrix} -\sin \theta & \cos \theta \end{bmatrix}^T$ perpendicular to the line segment connecting the rotors, and roll angular acceleration u_2 . We assume the system is subject to input constraints $\mathcal{U} = [(-40, -2\pi), (40, 2\pi)]$, gravitational acceleration a_g , as well as an unknown force due to wind. We assume this force affects acceleration in both the horizontal and vertical directions and is a function of altitude z . The resulting dynamics with normalized mass

and moment of inertia are

$$\begin{aligned}
\ddot{y} = \dot{v}_y &= -u_1 \sin \theta + g_1(z) \\
\ddot{z} = \dot{v}_z &= u_1 \cos \theta - a_g + g_2(z) \\
\ddot{\theta} = \dot{\omega} &= u_2
\end{aligned} \tag{3.19}$$

where g_1 and g_2 constitute the unknown wind forces in the horizontal and vertical directions, respectively. We again estimate g_1 and g_2 using GPs with a radial basis function kernel, initialized with several points around the starting point of the multirotor, and obtain high confidence bounds by considering posterior estimates up to three standard deviations from the mean. We set $h = 0.2$ s and collect an observation of $g_1(z)$ and $g_2(z)$ at every timestep.

The associated decomposition function takes the form

$$\delta(x, u, w, \hat{x}, \hat{u}, \hat{w}) = \left[v_y \quad d^{v_y} \quad v_z \quad d^{v_z} \quad \omega \quad u_2 \right]^T \tag{3.20}$$

$$\begin{aligned}
d^{v_y} &= -d^{b_1 b_2} \left(\begin{bmatrix} u_1 \\ d^{\sin}(\hat{\theta}, \theta) \end{bmatrix}, \begin{bmatrix} u_1 \\ d^{\sin}(\theta, \hat{\theta}) \end{bmatrix} \right) + w_1 \\
d^{v_z} &= d^{b_1 b_2} \left(\begin{bmatrix} u_1 \\ d^{\cos}(\theta, \hat{\theta}) \end{bmatrix}, \begin{bmatrix} u_1 \\ d^{\cos}(\hat{\theta}, \theta) \end{bmatrix} \right) - a_g + w_2
\end{aligned}$$

where $d^{b_1 b_2}$ is defined as before and d^{\sin}, d^{\cos} are the known tight decomposition functions for sin and cos, respectively (see [22, Equations 74–75]).

We define the safety set $\mathcal{X}_{\text{safe}} := [(-5, -30, -2, -30, -2\pi, -2\pi), (5, 30, 0, 30, 2\pi, 2\pi)]$ and task the system with entering the goal set $\mathcal{X}_{\text{goal}} := [(-5, -8, 9, -8, -\pi, -\pi), (5, 8, 11, 8, \pi, \pi)]$ while avoiding $\mathcal{X}_{\text{unsafe}}$, which is the union of a set of hyperrectangles (see the dashed red boxes in Figure 3.2). We again leverage Observation 1 and define our safety constraint as (Equation 3.16). We check that seven intermediate reachable sets between each timestep satisfy $[x(t), \hat{x}(t)] \subset$

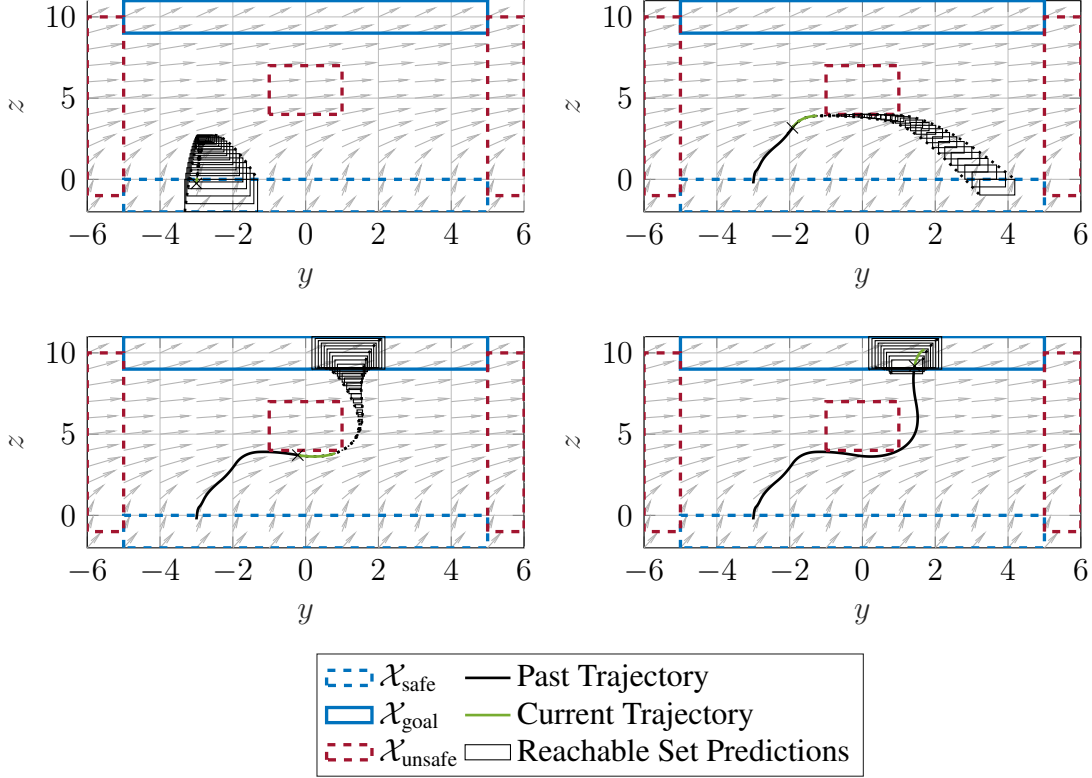


Figure 3.2: Execution of the Planar Multirotor case study. The arrows denote the unknown wind force acting on the system.

$(\mathcal{X}_{\text{unsafe}})^C$, and solve (Equation 3.8) for $D = 5$ timesteps.

Finally, we formulate our cost functions for each MPC scheme. We define our cost function for goal reaching as before with equation (Equation 3.17). If a feasible solution is found, the set of control actions is executed immediately, as these represent a control strategy that is guaranteed to end in $\mathcal{X}_{\text{goal}}$ with high probability. However, if this is infeasible, the controller attempts to explore the state space in a way that brings the rotor closer to the target altitude, while having a path back to safety. As g_1, g_2 are only functions of the altitude z , we can set altitude as the exploration metric $J_{k,\text{safe}}(\hat{R}_0, \dots, \hat{R}_D) = -\sum_{d=0}^D C_{d,2}$.

As shown in Figure 3.2, the system is initially unable to compute a control strategy that is guaranteed to reach the goal while avoiding the unsafe areas, so it reverts to exploring the state space (first and second plots). After sufficient exploration, the controller is able to guarantee with high probability that it reaches the goal (third and fourth plot). Thus, at all times, the controller has a

safe path to either $\mathcal{X}_{\text{safe}}$ or $\mathcal{X}_{\text{goal}}$ that holds with high probability. On average, it takes approximately nine minutes to compute the next control action single-threaded on a personal computer.

3.4.3 Boat on a River

We now consider a motorboat crossing a river where the exact flow behavior of the river is unknown. The position of the boat is (x, y) , the forward velocity of the boat is v , and the yaw angle is ψ . There exist two inputs: thrust produced by the motor u_1 , and rudder position u_2 . The resulting dynamics (adapted from [44]) are

$$\begin{aligned} \dot{v} &= -v + u_1 \\ \dot{\psi} &= 0.15vu_2 \\ \dot{x} &= v \cos \psi + g_x(x) \\ \dot{y} &= -v \sin \psi - f_y + g_y(y) \end{aligned} \tag{3.21}$$

where f_y is the known average flow of the river in the negative y direction and g_x and g_y denote the disturbed flow of the river in the x and y directions, respectively.

The associated decomposition function takes the form

$$\delta(x, u, w, \hat{x}, \hat{w}) = \left[d^v \quad d^\psi \quad d^x \quad d^y \right]^T \tag{3.22}$$

$$\begin{aligned} d^v &= -v + u_1 \\ d^\psi &= \begin{cases} 0.15 \min \{vu_2, \hat{v}u_2\}, & \hat{v} \geq v \\ 0.15 \max \{vu_2, \hat{v}u_2\}, & \hat{v} \leq v \end{cases} \\ d^x &= d^{b_1 b_2} \left(\begin{bmatrix} v \\ d^{\cos(\psi, \hat{\psi})} \end{bmatrix}, \begin{bmatrix} \hat{v} \\ d^{\cos(\hat{\psi}, \psi)} \end{bmatrix} \right) + w_x \end{aligned}$$

$$d^y = -d^{b_1 b_2} \left(\begin{bmatrix} \hat{v} \\ d^{\sin}(\hat{\psi}, \psi) \end{bmatrix}, \begin{bmatrix} v \\ d^{\sin}(\psi, \hat{\psi}) \end{bmatrix} \right) - f_y + w_y$$

where $d^{b_1 b_2}$ is defined as

$$d^{b_1 b_2}(b, \hat{b}) = \begin{cases} \min\{b_1 b_2, \hat{b}_1 b_2, b_1 \hat{b}_2, \hat{b}_1 \hat{b}_2\}, & \text{if } b \preceq \hat{b} \\ \max\{b_1 b_2, \hat{b}_1 b_2, b_1 \hat{b}_2, \hat{b}_1 \hat{b}_2\}, & \text{if } \hat{b} \preceq b. \end{cases} \quad (3.23)$$

and d^{\sin}, d^{\cos} are the known tight decomposition functions for sin and cos, respectively (see [22, Equations 74–75]).

We define the safety set $\mathcal{X}_{\text{safe}} := [(-5000, -6\pi, -2, -5), (5000, 6\pi, 1, 15)]$ and task the boat with crossing the river, i.e., reaching $\mathcal{X}_{\text{goal}} := [(-5000, -6\pi, 9, -5), (5000, 6\pi, 1, 12)]$. We also want the system to avoid several rocks that exist within the river, denoted by the red dashed hyperrectangles in Figure 3.3, as well as the riverbank, which we denote by the sets of states where $x < -2$ and $x > 12$. We check that five intermediate reachable sets between each timestep satisfy $[x(t), \hat{x}(t)] \subset (\mathcal{X}_{\text{unsafe}})^c$, and solve (Equation 3.8) for $D = 5$ timesteps.

Our cost functions for each MPC scheme are as follows. We define our cost function for reaching the goal as

$$J_{k,\text{goal}}(\hat{R}_0, \dots, \hat{R}_D) = \sum_{d=0}^D \|C_d - C_{\text{goal}}\|_2 \quad (3.24)$$

where C_d and C_{goal} denote the center points of the reachable and goal hyperrectangles, respectively, and we define our cost function for exploration as

$$J_{k,\text{safe}}(\hat{R}_0, \dots, \hat{R}_D) = \quad (3.25)$$

$$- \sum_{d=0}^D \left(\sigma_x(x_d) - \|x_d - x_{\text{goal}}\| e^{-\sigma_x(x_d)} \right)$$

$$- 0.5 \sum_{d=0}^D \left(\sigma_y(y_d) - \|y_d - y_{\text{goal}}\| e^{-\sigma_y(y_d)} \right)$$

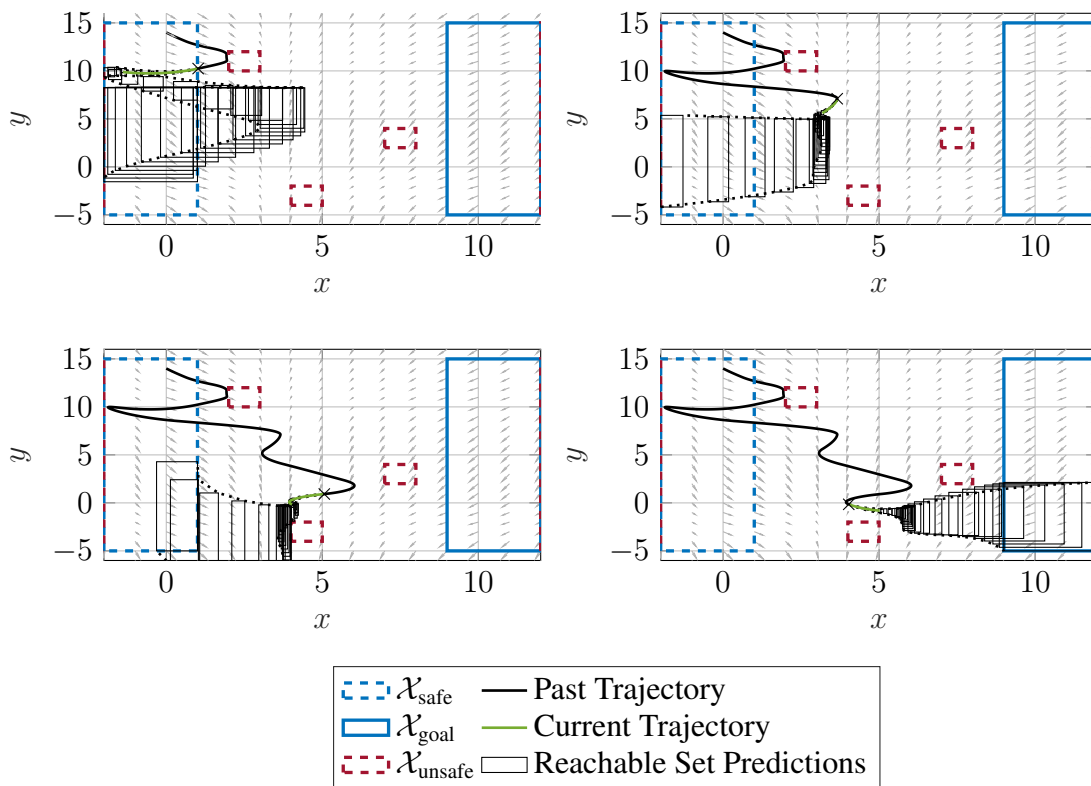


Figure 3.3: Execution of the river motorboat case study. The arrows denote the river flow acting on the system. Additionally, the set of states where $x < -2$ and $x > 12$ are considered part of $\mathcal{X}_{\text{unsafe}}$.

where $x_d, x_{\text{goal}}, y_d, y_{\text{goal}}$ denote the x and y center points of the reachable and goal hyperrectangles, and σ_x, σ_y denote the current estimated standard deviation of g_x, g_y . This makes the MPC solver prioritize information gain, similarly to the case study in [16], while biasing it toward states that bring the system closer to the goal. Multiplying the bias terms by $e^{-\sigma}$ allows the bias to be overcome if the expected information gain is high enough, and the exponential function is specifically chosen to mirror the structure of the GP radial basis kernel.

As shown in Figure 3.3, the system is initially unable to find a feasible control strategy that drives the boat to the goal while avoiding the unsafe areas. Thus, it reverts to exploring the river in order to gather observations of the river flow behavior. After sufficient exploration, the system is able to find a feasible strategy that ends in the goal region, at which point the algorithm executes that strategy and terminates. Therefore, the boat always has either a safe path back to the side of the river it started on, or a safe path to the other side.

CHAPTER 4

AN OPTIMISTIC APPROACH TO COST-AWARE PREDICTIVE CONTROL

4.1 Introduction

When the dynamics of a controlled system are not fully known, a common approach is to apply control actions to explore and observe the behavior of the system and adjust the control strategy as new information is collected. However, for systems with safety constraints that restrict allowable regions of the state space, the process of collecting observations must be designed to avoid unsafe behavior. Additionally, if there exists a separate objective that must be fulfilled, a balance must be struck between collecting observations of the unknown dynamics and progressing towards the objective.

Often, the system must optimize for some cost while in operation. For example, minimizing the magnitude of the input signal or the amount of power consumed by the system may be desirable. As collecting observations of the unknown behavior generally increases this cost, a natural strategy is to collect the minimum number of observations needed to guarantee safety and objective fulfillment, then immediately drive the system to the goal. However, as the disturbance signal is state-based, it may be that unexplored areas of the state space would have incurred a lower disturbance and a lower overall cost. Consequently, a key challenge arises in determining the optimal tradeoff between exploring the state space and exploiting the current least-cost path to the goal.

As an example, consider a planar quadrotor operating in an unknown wind field (see Figure 4.1-Figure 4.2), where the quadrotor must fly to the goal area while minimizing energy usage. Given some observations of the wind field at lower altitudes, it is possible to calculate a control strategy that arrives at the goal with high probability. Since the wind is observed to be blowing against the quadrotor at this altitude, the overall cost (i.e. the energy usage) is likely to be quite high. However, the wind at higher altitudes is less strong, and even blowing in the direction of the goal.

If the quadrotor were to discover this, it could achieve lower energy usage. Exploring the windfield is, however, not without risk as the quadrotor needs to spend energy to fly higher and collect these observations. Thus, the objective is to design a control scheme that quantifies the risk and expected reward of exploration, and determines whether it is worth exploring to collect more observations.

In this chapter, we consider systems in continuous-time subject to state-dependent unknown components that enter the dynamics nonlinearly. We leverage the mixed monotonicity property of dynamical systems (see [1] for an overview) and utilize previous results [22] to obtain hyperrectangular overapproximations of the reachable sets of the system that hold with high probability. These overapproximations are obtained by computing a single trajectory of an appropriately constructed *embedding system* that is an ordinary differential equation with twice the dimension of the original system.

Comparing to existing approaches, we consider continuous time systems with nonlinear disturbances and we use reachability techniques that are computationally efficient and scalable, as demonstrated on a multirotor case study with six states. Moreover, this approach avoids excessive conservatism that often occurs when linearizing the dynamics and outerbounding the linearization error using the Lipschitz constant of the dynamics [21].

We build upon the work presented in the previous chapter and formulate a control scheme that considers potentially lower-cost trajectories, and provide an analysis of a novel strategy for selecting when to pursue said trajectories. Specifically, we allow the MPC solver to *directly* adjust the probabilistic bounds on the disturbance so that it can consider lower-cost potential trajectories. We offer theoretical results that show the advantages of this novel strategy on a simplified system, and empirical results demonstrating performance on a more complex system. Complete results for general nonlinear systems are left as future work.

The rest of this chapter is organized as follows: in section 4.2, we cover related work, before formally defining our problem in section 4.3. We then outline the modifications made to enable speculation on lower-cost trajectories as well as a theoretical analysis on a simplified system in section 4.4, and perform an experimental analysis of each in section 4.5.

4.2 Related Work

Our work is most closely related to [16], which presents a discrete time MPC formulation that provides high probability safety guarantees in the presence of uncertain dynamics. The paper [16] also uses GPs to estimate the unknown dynamics, and then high probability ellipsoidal overapproximations of reachable sets are computed by combining these estimations with a linearization of the dynamics, where the error is bounded using Lipschitz constants. We draw from the problem setup proposed in [16] and consider a nonlinear dynamical system whose dynamics are not fully known. As in [16], we estimate the unknown component using GP regression. Exploration of the state space is allowed so long as a feasible return trajectory is available that returns the system to a known safe set.

Similarly, [39] provides safety guarantees on reinforcement learning for robotic applications by learning the system’s unknown dynamics using GPs, then employing Hamilton-Jacobi-Isaacs (HJI) reachability analysis to iteratively update the safety set of the system. The authors in [32] also provide safety guarantees (defined in terms of stability guarantees) on model-based reinforcement learning using Lyapunov-based stability verification. Additionally, [45] proposes an exploration/exploitation reachability-based control framework utilizing Bayesian meta-learning to learn the entirety of the dynamical model, while [46] uses Lipschitz interpolation to calculate reachable sets towards the same end. Alternatively, [47] proposes a Bayesian MPC algorithm wherein the model predictive controller optimizes directly on the parametric model derived from collected samples to enable the exploration/exploitation tradeoff, though safety constraints are not explicitly considered.

The paper [40] presents an adaptive MPC framework under state-dependent uncertainty. Safety is guaranteed by approximating the graph of the uncertainty via envelopes defined by quadratic constraints. A set of convex optimization problems is solved to guarantee robust constraint satisfaction for all possible values of system uncertainty. However, a key assumption of [40] is that the uncertainty is additive and globally Lipschitz with a known Lipschitz constant.

The work [32] presents a learning algorithm that explicitly considers safety defined in terms of Lyapunov stability guarantees, [19] proposes a general safety framework based on Hamilton-Jacobi reachability methods, [33, 34, 35] synthesize control barrier functions online to guarantee safety, and [36] achieves safety by estimating the Lipschitz constant of the disturbance. Other works, such as [37, 38, 39], explore learning and updating safety sets in an online manner. For MPC-based approaches, proposed frameworks are robust to uncertainty by, e.g., assuming a known Lipschitz constant [40], assuming the uncertainty is parametric [41], or applying MPC to iterative learning control [42].

In the realm of optimizing control strategies in the face of uncertainty, [48] presents a gradient descent algorithm that simultaneously learns and optimizes for the partially unknown dynamics of a discrete-time system, [49] derives a set of sampling point selection strategies that result in data-efficient learning of an unknown Gaussian Process system, and [50] leverages a Neural Control Contraction Metric to ensure safety of a system while exploring and observing state-dependent uncertainties.

In our previous work [22], we derive high probability bounds on the unknown disturbance behavior by modeling it as a GP. In turn, these bounds enable us to calculate overapproximations of the reachable sets of the system that hold with high probability. We then use these reachable set overapproximations in an MPC formulation to compute a control strategy that is safe with high probability [51]. We accomplish this by requiring that the reachable set overapproximations never intersect the unsafe regions of the state space. This results in a control strategy that is not only safe with high probability, but is also able to ensure fulfillment of objectives with high probability.

Our approach differs from these existing works in several key aspects. Specifically, we leverage mixed monotonicity from nonlinear systems theory to calculate the high probability reachable set overapproximations. This allows us to work in continuous time, incorporate the uncertainty nonlinearly, avoid excess conservatism that results from linearizing the dynamics and outerbounding the linearization error, and provide tighter bounds than outerbounding the uncertainty through Lipschitz constants. Additionally, this enables our method to scale to systems of higher dimension,

something which has traditionally been a challenge for other reachability-based techniques, though this is an active area of research as shown in [20, 52] where warm-starting is employed to speed up computation times in HJI-based methods.

The outlined strategy also bears a resemblance to traditional closed-loop MPC, but there are a few key distinctions. Chiefly, the computed strategy is still open-loop. Adjusting the confidence bounds in the described manner acts as a proxy to closing the loop around the disturbance input, as it is implicit in this strategy that an observation of the disturbance behavior will be collected and the control actions will be recomputed at the next time step. As a result, the strategy gains some of the forward-looking benefits of closed-loop MPC while avoiding the need for a computationally complex dynamic programming solution.

The main novelty of this work is that we explicitly consider the tradeoff between safety and performance. Most of the existing literature focuses on providing safety guarantees during the learning process, assuming that the system has enough time and resources to reach the optimal policy. By contrast, we present a formulation wherein the desired probability of safety is a tunable parameter, allowing for the balance between safety and performance, given limited time and resources, to be fully customizable.

4.3 Problem Setup

As in the previous chapter, we consider the continuous-time nonlinear dynamical system

$$\dot{x} = f(x, u, w) \tag{4.1}$$

with f continuously differentiable where $x \in \mathbb{R}^n$ is the system state, $u \in \mathcal{U} \subset \mathbb{R}^m$ is the input constrained to take values in \mathcal{U} , and $w \in \mathbb{R}^p$ is an unknown, state-dependent component of the dynamics so that $w_i = g_i(x)$ where g_i is unknown. Throughout, we assume the input constraint set has the form $\mathcal{U} = [\underline{u}, \bar{u}]$ for some $\underline{u}, \bar{u} \in \mathbb{R}^m$, $\underline{u} \preceq \bar{u}$, that is, \mathcal{U} is a hyperrectangle defined by corners \underline{u} and \bar{u} .

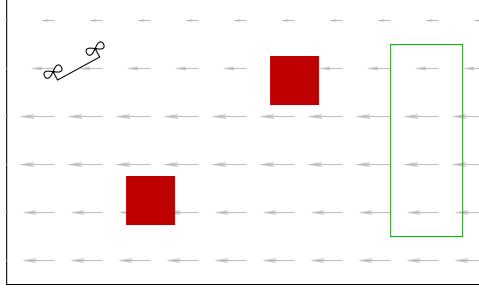


Figure 4.1: An illustrative example system that fits the problem setting. A planar multirotor must fly to the goal region (green) while avoiding obstacles in midair (red). There is also a wind force acting on the multirotor which varies based on its location and is unknown a priori. Observations of this force can be collected, and the objective is to guarantee a safe path to the goal, potentially while minimizing the energy spent.

Given a feedback control strategy $u = \pi(t, x)$, we denote by $\phi(t, x_0, \pi)$ the resulting true closed-loop state trajectory of (Equation 4.1) when $w = g(x)$ and the system is initialized at x_0 at time 0. If π is time-invariant, we write $\pi(x)$ instead. Additionally, given some $X_0 \subseteq \mathbb{R}^n$, the *T-horizon reachable set from X_0* for (Equation 4.1) is the set of states reachable over the time horizon T from any initial condition $x_0 \in X_0$ and is denoted

$$R(T, X_0, \pi) = \{\phi(T, x_0, \pi) \mid x_0 \in X_0\}. \quad (4.2)$$

Our objective, slightly modified from the previous chapter, is to steer the system to a goal region with minimal cost. For example, given a planar multirotor operating in a wind field as in Figure 4.1, the objective is to avoid crashing into the mid-air obstacles while trying to reach the other end of the state space, while minimizing the amount of power used. The control approach proposed in the previous chapter involves incrementally making progress towards the goal while learning the unknown component of the dynamics and ensuring the system can always safely return to $\mathcal{X}_{\text{safe}}$ if needed, until it can be guaranteed that the system can safely reach the goal. While moving towards the goal, the system is able to collect information about its dynamics and reduce the uncertainty in its estimate of $g(x)$, allowing it more freedom to safely explore. However, we now consider that the system must optimize for some cost $J(x, u)$ (e.g. fuel consumption) while in operation.

System Behavior

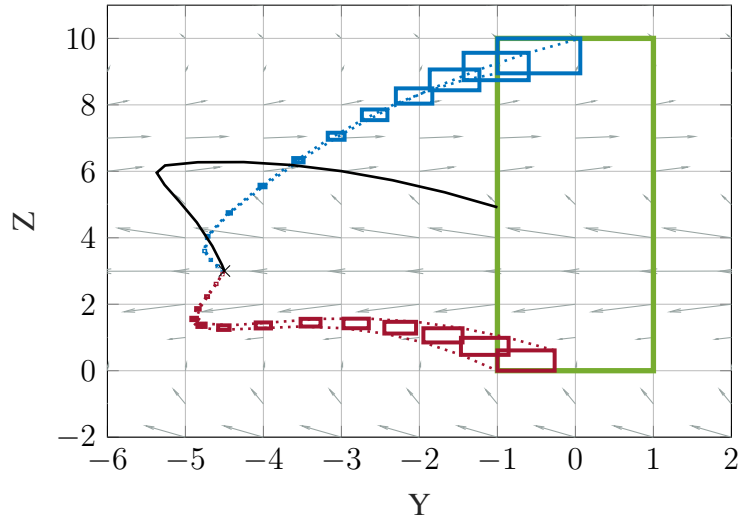


Figure 4.2: For systems in which there is a cost to be minimized (i.e., energy consumption), such as the planar multirotor operating in an unknown wind field shown above, the disturbance behavior in unexplored areas of the state space may incur a lower overall cost. As shown, the system tries to reach the goal region (green rectangle) while minimizing the energy spent. The multirotor only has a few observations of the wind around its starting location, thus considering the worst-case behavior of the disturbance (Pessimism) results in the red trajectory. However, allowing the multirotor to adjust the estimated worst-case bounds (Optimism) allows the multirotor to explore, resulting in the blue trajectory. In this case, it is advantageous to be Optimistic, as the blue trajectory is closer to the calculated optimal trajectory in black. These trajectories were generated from an execution of the second case study in section 4.5.

Thus, a strategy must be developed to determine when exploration, which generally increases this cost, is worth the potential long-term gains while fulfilling the objective. We formalize this in the following problem statement.

Problem 4. Consider a system as in (Equation 4.1) with specified initial condition x_0 and input constraints \mathcal{U} . Given a goal region $\mathcal{X}_{\text{goal}} \subset \mathbb{R}^n$, the objective is to compute a feedback control strategy $u = \pi(t, x)$ that reaches the goal with lower expected cost than the nominal strategy that solves Problem 3.

This problem setup assumes that it is possible for exploration to lower the incurred cost; thus, we only consider scenarios in which the unknown disturbance behavior has at least an indirect effect on the overall cost. Additionally, for this chapter, we only consider cost incurred and no

longer assume the existence of unsafe sets in this problem, which also removes the need for a safety set.

An example of this setup can be found in Figure 4.2, wherein a planar multirotor is attempting to fly to the goal with minimal energy expenditure. The nominal strategy that solves Problem 3 produces the red trajectory, whereas the trajectory that would be optimal if wind disturbance were exactly known is in black. The objective of Problem 4 is, given current knowledge of the disturbance behavior, to produce strategies akin to the blue trajectory, which is closer to the optimal trajectory than the nominal red trajectory. We return to this planar multirotor system with a case study in section 4.5.

4.4 An Optimistic Control Algorithm

Our main contribution of this chapter is to modify the control scheme outlined in the previous chapter (see algorithm 1), with the goal of solving Problem 4. The key insight is that the values of β are typically fixed to guarantee the probability of bounding the disturbance behavior defined by $1 - \eta$. As outlined in section 3.3, this allows for safety guarantees that hold with high probability. However, these high probability guarantees often result in conservative control actions that leads to suboptimal cost minimization.

Thus, adjusting these bounds allows us to calculate alternative trajectories that hold with lower probability. In other words, we allow the system more freedom to explore cost-minimizing paths by reducing the required amount of conservatism in estimating the disturbance behavior.

We modify the embedding system to accept the desired level of confidence as an input, resulting in

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = e(x, \hat{x}, u, \underline{\beta}, \bar{\beta}) := \begin{bmatrix} \delta(x, u, \underline{\gamma}_{\underline{\beta}}(x, \hat{x}), \hat{x}, \bar{\gamma}_{\bar{\beta}}(x, \hat{x})) \\ \delta(\hat{x}, u, \bar{\gamma}_{\bar{\beta}}(x, \hat{x}), x, \underline{\gamma}_{\underline{\beta}}(x, \hat{x})) \end{bmatrix} \quad (4.3)$$

where $\underline{\gamma}_{\underline{\beta}}$ and $\bar{\gamma}_{\bar{\beta}}$ represent the bounding functions (Equation 2.58) and (Equation 2.59) with their values of β_t set to $\underline{\beta}$ and $\bar{\beta}$, respectively.

We then sample this new embedding system with timestep h such that at each step $k = t/h$,

$$\begin{bmatrix} x[k+1] \\ \hat{x}[k+1] \end{bmatrix} = \Phi^e(h; (x[k], \hat{x}[k]), \pi_k, \underline{\beta}, \bar{\beta}), \quad (4.4)$$

where π_k is again the zero-order hold controller applied from time kh to $(k+1)h$. Below, we assume π_k is a constant policy $\pi_k(t, x) \equiv u_k$ for some $u_k \in \mathcal{U}$ to be designed by an MPC scheme. Thus, taking $\hat{R}_k = [x[k], \hat{x}[k]]$ overapproximates the reachable set of (Equation 4.1) at time $t = kh$ with high probability. These discretized reachable sets are then included in the MPC as follows:

$$\underset{\Pi, \underline{\beta}, \bar{\beta}}{\text{minimize}} \quad J_k(\hat{R}, \Pi) \quad (4.5)$$

subject to:

$$\text{(Equation 4.4), } x[0] = \hat{x}[0] \text{ given, } u_d \in \mathcal{U} \quad \forall d \in \{0, \dots, D-1\}$$

$$\hat{R}_d = [x[d], \hat{x}[d]] \quad \forall d \in \{0, \dots, D\}$$

$$\hat{R}_D \subset \mathcal{X}_{\text{obj}}$$

$$\underline{\beta}, \bar{\beta} \in [\beta_{\text{MIN}}, \beta_{\text{MAX}}],$$

$$P(\underline{\gamma}_{\underline{\beta}} \preceq g(x) \preceq \bar{\gamma}_{\bar{\beta}}) \geq 1 - \eta_o$$

where \mathcal{X}_{obj} is the goal hyperrectangle, and $J_{k,\text{obj}}$ is the desired cost function. The value of β_{MAX} is the value of $\sqrt{\beta_t}$ where β_t is defined as in [22, Theorem 7], which results in bounding functions that encapsulate the disturbance behavior with probability at least $1 - \eta$, where η represents the same value as in section 3.3. The value of β_{MIN} where $\beta_{\text{MIN}} \in [0, \beta_{\text{MAX}}]$ is a user-defined value. Additionally, $1 - \eta_o$ where $1 - \eta_o \in [0, 1 - \eta]$ is a user-defined value that determines the minimum probability desired for the bounds to encapsulate the disturbance behavior. This chance constraint is imposed by converting the chosen $\underline{\beta}, \bar{\beta}$ using the associated z-score to the resulting probability value based on the Gaussian distribution. Thus, when this problem is feasible, it produces the lowest-cost set of inputs that is guaranteed to drive the system into the objective with probability

at least $1 - \eta_o$.

We then note that the nominal strategy of fixing $\underline{\beta}, \bar{\beta} = \beta_{\text{MIN}} = \beta_{\text{MAX}}$ recovers the control strategy that solves Problem 3. While this solution provides feasible trajectories, it effectively assumes the worst-case disturbance behavior and as a result may be overly conservative and thus incur more cost than is necessary. Thus, moving forward we refer to this strategy as Pessimistic. This strategy is outlined in algorithm 2.

Algorithm 2 Pessimistic Control Strategy

Data: Embedding system (Equation 4.3) sampled as (Equation 4.4), bounding functions $\underline{\gamma}_{\beta}, \bar{\gamma}_{\beta}$

for $k = 0, 1, \dots$ **do**

$(\Pi, \underline{\beta}, \bar{\beta}) \leftarrow$ solve MPC (Equation 4.5), $\beta_{\text{MIN}} = \beta_{\text{MAX}}$;
 $\Pi_k \leftarrow \Pi$;
 $x_{k+1} \leftarrow$ apply $u(t) = \Pi_{k,0}(x(t))$ to (Equation 4.1) until $t = (k + 1)h$;
 collect observation and update $\underline{\gamma}_{\beta}, \bar{\gamma}_{\beta}$

Our proposed strategy is to allow the MPC to modify the bounds by setting $\beta_{\text{MIN}} = 0$ and selecting a probability η_o such that $1 - \eta_o \leq 1 - \eta$, thereby expanding the search space of feasible trajectories available to the solver. While allowing the calculated bounds to shrink means that the resulting bounds have a lower probability of encapsulating the disturbance, the resulting exploration and new observations allow the system to take advantage of the disturbance behavior in areas that further decrease the cost incurred. Selecting $1 - \eta_o > 0$ additionally preserves some of the robustness of Pessimism, by accounting for the fact that new observations are going to be collected and allowing the MPC scheme to “look ahead” despite being open-loop. As the trajectories produced are essentially speculation, we refer to this strategy moving forward as Optimistic. This strategy is outlined in algorithm 3.

An example of the different trajectories produced by each strategy is shown in Figure 4.2 for the planar multirotor system described in section 4.5. The system has a few observations of the disturbance in the range $Z \in [0, 4]$. The red (Pessimistic) trajectory assumes worst-case bounds and thus prioritizes exploitation of the known disturbance behavior, while the blue (Optimistic) trajectory assumes tighter confidence bounds, though they may not correctly encapsulate the dis-

Algorithm 3 Optimistic Control Strategy

Data: Embedding system (Equation 4.3) sampled as (Equation 4.4), bounding functions $\underline{\gamma}_\beta, \overline{\gamma}_\beta$

for $k = 0, 1, \dots$ **do**

$(\Pi, \underline{\beta}, \overline{\beta}) \leftarrow$ solve MPC (Equation 4.5), $\beta_{\text{MIN}} = 0$;
 $\Pi_k \leftarrow \Pi$;
 $x_{k+1} \leftarrow$ apply $u(t) = \Pi_{k,0}(x(t))$ to (Equation 4.1) until $t = (k + 1)h$;
 collect observation and update $\underline{\gamma}_\beta, \overline{\gamma}_\beta$

turbance behavior. Solving the optimal control problem for the true disturbance behavior results in the black trajectory.

In the next sections, we provide a proof that the Optimistic strategy incurs lower expected cost than the Pessimistic strategy for a simplified system, as well as empirical results showing the same on a higher-dimensional quadrotor system.

4.4.1 Theoretical Results from a Simplified Setting

The above algorithm applies to general nonlinear systems in continuous-time and, as we demonstrate in the following sections, its benefits are supported empirically. However, its generality prevents provable guarantees. In this subsection, we explore a simplified setting under which theoretical guarantees are available. These results provide a degree of justification to the empirical successes that follow in section 4.5.

Consider the discrete-time system

$$x[k + 1] = x[k] + u[k] + w \quad (4.6)$$

with state x , input u , and disturbance $w = g(x)$ for some unknown function $g(x)$, from which observations can be drawn. We task the controller with driving the system into a goal region $X_G \in [\underline{x}_G, \overline{x}_G]$ within two timesteps while minimizing the total input used, i.e. we want to solve

$$\underset{u[0], u[1]}{\text{minimize}} \quad |u[0]| + |u[1]| \quad (4.7)$$

subject to:

(Equation 4.6), $x[0]$ given,

$$x[1] \in X_G \parallel x[2] \in X_G.$$

At $k \in \{0, 1\}$, we assume that we can noiselessly observe $g(x[k])$. Following the approach of section 4.4, we form the associated embedding system of (Equation 4.6) and insert the appropriate confidence bounds on $g(x)$, resulting in

$$\begin{bmatrix} x[k+1] \\ \hat{x}[k+1] \end{bmatrix} = \begin{bmatrix} x[k] + u[k] + \underline{\gamma}_{\underline{\beta}}(x[k], \hat{x}[k]) \\ \hat{x}[k] + u[k] + \overline{\gamma}_{\overline{\beta}}(x[k], \hat{x}[k]) \end{bmatrix} \quad (4.8)$$

Thus, as a proxy to (Equation 4.7), we solve

$$\underset{u[0], u[1], \underline{\beta}, \overline{\beta}}{\text{minimize}} \quad |u[0]| + |u[1]| \quad (4.9)$$

subject to:

(Equation 4.8), $x[0] = \hat{x}[0]$ given,

$$[x[1], \hat{x}[1]] \subseteq X_G \text{ or } [x[2], \hat{x}[2]] \subseteq X_G$$

$$\underline{\beta}, \overline{\beta} \in [\beta_{\text{MIN}}, \beta_{\text{MAX}}].$$

Note that, in this case, we impose no minimum probability requirement; as there are only two timesteps, the advantage of “looking ahead” is minimal.

Theorem 9. *Given the optimization problem (Equation 4.7), the Pessimistic strategy of solving (Equation 4.9) with $\underline{\beta} = \overline{\beta} = \beta_{\text{MAX}}$ incurs greater expected cost than the Optimistic strategy of solving (Equation 4.9) allowing $\underline{\beta}, \overline{\beta} \in [0, \beta_{\text{MAX}}]$.*

Proof. We note that for this system, as we impose no minimum probability requirement, an option that is always available to the Optimistic strategy is to set $\underline{\beta} = \overline{\beta} = 0$, effectively modeling the disturbance directly by the mean of the GP produced by the available observations. Moving

forward, we assume that the Optimistic strategy always exercises this option, as any trajectory that is feasible for the Optimistic strategy when $\underline{\beta}, \overline{\beta} \neq 0$ is also feasible when $\underline{\beta} = \overline{\beta} = 0$. Generally speaking, it is safe to assume that the Optimistic strategy reduces the bounds to the minimum probability requirement, as there is never a disadvantage in doing so.

Denote by $u_O^j[k], u_P^j[k]$ the control input proposed by the Optimistic and Pessimistic strategies, respectively, for timestep k calculated at timestep j , and denote by $u_O[k], u_P[k]$ the actual inputs applied by each strategy at timestep k . Similarly, we utilize $x_{\{O,P\}}^j[k], \hat{x}_{\{O,P\}}^j[k]$ to denote the proposed reachable set hyperrectangle endpoints calculated by each strategy at timestep j for timestep k , and $x_{\{O,P\}}[k]$ to denote the actual state encountered by each strategy at timestep k . Finally, we utilize $J_{\{O,P\}}^j$ to denote the cost of the proposed control actions of each strategy calculated at timestep j , and $J_{\{O,P\}}$ to denote the actual incurred cost of each strategy.

First, we note that, given a feasible Pessimistic strategy $u_P^0[0], u_P^0[1]$ with resulting reachable set approximations $[x_P^0[1], \hat{x}_P^0[1]], [x_P^0[2], \hat{x}_P^0[2]]$, it must hold that

$$\begin{aligned} (|u_P^0[0]| + |u_P^0[1]|) - (|u_O^0[0]| + |u_O^0[1]|) & \geq \max \{ \sigma(x_P^0[1])\beta_{\text{MAX}}, u_P^0[1] \}, \end{aligned} \quad (4.10)$$

or, equivalently,

$$J_P^0 - J_O^0 \geq \max \{ \sigma(x_P^0[1])\beta_{\text{MAX}}, |u_P^0[1]| \} \quad (4.11)$$

as the Optimistic strategy can simply take the feasible control actions chosen by the Pessimistic strategy, and trim the excess input proposed by that strategy needed to overcome the uncertainty that Pessimism works with. For example, if $\mu(x_P^0[1])$ is such that

$$x_P^0[1] + \mu(x_P^0[1]) \leq \underline{x}_G, \quad (4.12)$$

the Pessimistic strategy proposes

$$u_P^0[1] = \underline{x}_G - (x_P^0[1] + \mu(x_P^0[1]) - \sigma(x_P^0[1])\beta_{\text{MAX}}) \quad (4.13)$$

while the Optimistic strategy may propose

$$u_O^0[1] = \underline{x}_G - (x_P^0[1] + \mu(x_P^0[1])). \quad (4.14)$$

We then note that, per the problem setup, it holds that

$$u_{\{O,P\}}^1[1] = u_{\{O,P\}}[1] \quad (4.15)$$

as at timestep $k = 1$, the strategies have perfect knowledge of $g(x)$ at their current state and thus knows exactly the value of $u[1]$ needed to arrive at the goal. Similarly, as the strategies have perfect knowledge of $g(x[0])$ at timestep $k = 0$, it holds that

$$x_{\{O,P\}}^0[1] = x_{\{O,P\}}[1] \quad (4.16)$$

Thus, the actual incurred cost of any given strategy is equivalent to

$$J_{\{O,P\}} = |u_{\{O,P\}}^0[0]| + |u_{\{O,P\}}^1[1]|. \quad (4.17)$$

We then note that, as the Optimistic strategy estimates the disturbance using $\mu(x)$ only, it holds that

$$E(u_O^0[1] - u_O^1[1]) = 0 \quad (4.18)$$

which consequently means that

$$E(J_O) = J_O^0 \leq J_P^0 - \max\{\sigma(x_P^0[1])\beta_{\text{MAX}}, |u_P^0[1]|\} \quad (4.19)$$

Finally, we note that the Pessimistic strategy has a limit to how much it can improve upon observing $g(x_P[1])$ if it has correctly bounded the disturbance behavior (i.e. $g(x_P[1]) \in [\mu(x_P[1]) - \sigma(x_P^0[1])\beta_{\text{MAX}}, \mu(x_P[1]) + \sigma(x_P^0[1])\beta_{\text{MAX}}]$): $E(|u_P^0[1]| - |u_P^1[1]|) \leq \max\{\sigma(x_P^0[1])\beta_{\text{MAX}}, |u_P^0[1]|\}$. If the Pessimistic strategy has incorrectly bounded the disturbance behavior, then $E(|u_P^0[1]| - |u_P^1[1]|) = 0$, as the Pessimistic bounds are formulated by taking the same deviation from the mean towards either side.

Thus, for the Pessimistic strategy,

$$E(|u_P^0[1]| - |u_P^1[1]|) \leq \max\{\sigma(x_P^0[1])\beta_{\text{MAX}}, |u_P^0[1]|\}. \quad (4.20)$$

As a result,

$$E(J_P) \geq J_P^0 - \max\{\sigma(x_P^0[1])\beta_{\text{MAX}}, |u_P^0[1]|\}. \quad (4.21)$$

Combining (Equation 4.19) and (Equation 4.21) gives

$$E(J_O) \leq E(J_P). \quad (4.22)$$

■

Thus, the Optimistic Strategy outlined solves Problem 4.

While this theoretical result is only proven for the system (Equation 4.6), in the next section we provide empirical results that show that the Optimistic strategy outperforms Pessimism on a higher-dimensional system, suggesting that this result is applicable to general systems.

4.5 Case Study Results

In this section we provide a case study of a Monte Carlo simulation comparing the Optimistic and Pessimistic strategies outlined in section 4.4.¹ We turn to the case study outlined in section 4.1

¹A code repository for reproducing this case study is available at https://github.com/gtfactslab/Cao_OptimisticControl.

of a planar quadrotor flying in an unknown wind field. In this system, the horizontal and vertical position of the multirotor are denoted y and z , and the roll angle is denoted θ . There are also two inputs: thrust u_1 acting at the center of mass in the direction $\begin{bmatrix} -\sin \theta & \cos \theta \end{bmatrix}^T$, and the roll angular velocity u_2 . Thus, the resulting dynamics with normalized mass and moment of inertia are

$$\begin{aligned}\ddot{y} &= -u_1 \sin \theta + g_1(z) \\ \ddot{z} &= u_1 \cos \theta - a_g + g_2(z) \\ \dot{\theta} &= u_2\end{aligned}\tag{4.23}$$

where g_1 and g_2 constitute the unknown wind forces in the horizontal and vertical directions, respectively. For this case study, we note that both functions g_1 and g_2 are only dependent on z ; this is a deliberate choice to illustrate the risk/reward tradeoff, as flying to different altitudes will naturally incur more cost in the short term with the hope of reducing cost in the long term.

The associated decomposition function takes the form

$$\delta(x, u, w, \hat{x}, \hat{w}) = \begin{bmatrix} v_y & d^{v_y} & v_z & d^{v_z} & \omega & u_2 \end{bmatrix}^T\tag{4.24}$$

$$\begin{aligned}d^{v_y} &= -d^{b_1 b_2} \left(\begin{bmatrix} u_1 \\ d^{\sin}(\hat{\theta}, \theta) \end{bmatrix}, \begin{bmatrix} u_1 \\ d^{\sin}(\theta, \hat{\theta}) \end{bmatrix} \right) + w_1 \\ d^{v_z} &= d^{b_1 b_2} \left(\begin{bmatrix} u_1 \\ d^{\cos}(\theta, \hat{\theta}) \end{bmatrix}, \begin{bmatrix} u_1 \\ d^{\cos}(\hat{\theta}, \theta) \end{bmatrix} \right) - a_g + w_2\end{aligned}$$

where $d^{b_1 b_2}$, d^{\sin} , and d^{\cos} are defined as before. Thus, with this decomposition function we can craft our associated embedding system and MPC schemes accordingly.

We task the quadrotor with travelling to a known objective region while minimizing the sum of the absolute value of the forces F_1, F_2 applied to the system through the propellers; these are

easily derived from u_1 and u_2 by solving the linear set of equations

$$\begin{bmatrix} 1 & 1 \\ \frac{L}{2} & -\frac{L}{2} \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad (4.25)$$

where L is the distance from the propeller to the center of mass of the multirotor.

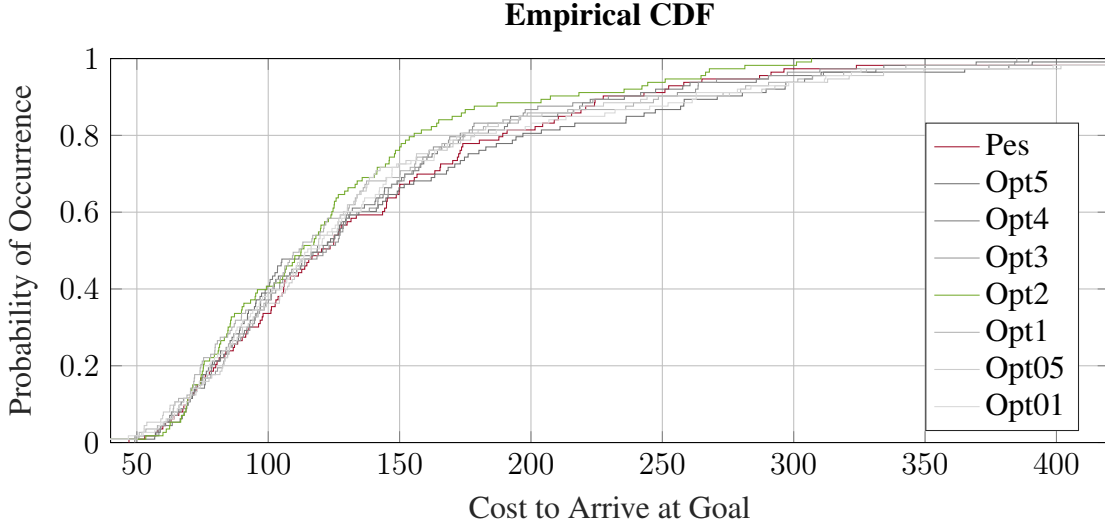


Figure 4.3: Empirical cumulative distribution function (cdf) of the incurred cost of each strategy over 113 total runs. Each curve represents the overall results of each strategy executed in the Monte Carlo simulation. The horizontal axis denotes the cost incurred to arrive at the goal, while the vertical axis denotes the proportion of runs which incurred that cost or lower. The closer to the left a curve is, the better. For example, the Opt2 Strategy (green) incurred a cost of 150 or lower in approximately 80% of its runs. From these results, we can see that there is a tradeoff. Compared to the Pessimistic strategy (red), the Optimistic Strategies (gray) had a larger proportion of runs incur a cost of 200 or lower. However, they generally had a *smaller* proportion of runs incur a cost of 250 or lower. Only one strategy, Opt2 (green), which has a minimum probability $1 - \eta = 0.2$, unambiguously outperforms Pessimism. The respective minimum probabilities $1 - \eta$ of each strategy are outlined in Table 4.1.

Thus, our final cost function becomes

$$J(\Pi) = \sum_{d=0}^{D-1} |F_{1,d}| + |F_{2,d}|. \quad (4.26)$$

We perform a Monte Carlo simulation, varying the underlying disturbance functions g_1 and g_2 for each iteration, and initializing the system with a few observations. We then simulate each

Table 4.1: Statistical summary of the incurred cost of the Pessimistic strategy as well as the Optimistic strategies tested with different minimum required probabilities.

Strategy	$1 - \eta$	Mean	Std. Dev.
Pessimism	≥ 0.99	140.97	77.88
Opt5	≥ 0.5	144.72	85.69
Opt4	≥ 0.4	136.20	70.70
Opt3	≥ 0.3	136.34	70.89
Opt2	≥ 0.2	124.44	57.60
Opt1	≥ 0.1	136.50	81.89
Opt05	≥ 0.05	139.72	85.01
Opt01	≥ 0.01	140.21	84.11

strategy and record the total cost incurred once the multirotor has reached the objective. Figure 4.2 showcases an instance of the Monte Carlo simulation and the resulting trajectories produced by some of the strategies.

We plot the resulting empirical cumulative distribution function (cdf) of the incurred cost of each strategy over 113 runs in Figure 4.3 and provide statistics in Table 4.1. Overall, the Optimistic strategies tended to outperform the Pessimistic strategy, though as can be seen, the minimum probability requirement affects the performance of the Optimistic strategy. It is especially important to note that there seems to be an ideal tradeoff; the lowest minimum probability does not necessarily translate to the lowest incurred cost. This is in line with the idea that a nonzero minimum probability η_o gives the MPC scheme the ability to “look ahead” while retaining some robustness.

These results suggest that Optimism is a step in the right direction towards deriving the optimal theoretical strategy. Allowing the controller to determine the best-case disturbance bounds while still ensuring it keeps known observations in mind is a good tradeoff between exploration and exploitation, and the minimum probability requirements can be tweaked as necessary.

CHAPTER 5

SAFE AND PERFORMANT CONTROL VIA EFFICIENT OVERAPPROXIMATION OF THE REACHABLE SET PROBABILITY DISTRIBUTION

5.1 Introduction

Control of systems with partially unknown dynamics often requires compromising performance to satisfy safety guarantees. This is generally because worst-case disturbance behavior must be considered in order to maintain these guarantees. To achieve greater performance, it is advantageous to consider not only the worst-case disturbance but also the overall distribution of possible disturbance behavior.

For linear systems, it is possible to approximate the effects of a disturbance modeled using, e.g., a Gaussian distribution by evolving key parameters through the system and then reconstituting the distribution to maintain safety [16]. However, for nonlinear systems, especially systems that are nonlinear in the disturbance input, these nonlinearities distort the impact of the disturbance behaviors through the system. Motivated by this, we develop a method for overapproximating multiple probability levels of the reachable sets of the system. This allows us to capture the effects of the nonlinearities on the disturbance behavior and utilize them in an optimization framework for achieving greater performance while maintaining safety.

Reachability-based methods are generally well-suited for the problem of ensuring safe control of partially unknown systems. For example, [16] achieves safety via ellipsoidal overapproximations of the linearized system that account for the unknown dynamics via GPs, and [19] utilizes Hamilton-Jacobi reachability tools to develop a general safety framework for uncertain robotic systems. Alternatively, [53] develops a linearization-based reachability overapproximation technique for systems with unknown parameters and inputs which is demonstrated in a collision avoidance scenario for autonomous vehicles in [54]. In [39], the authors use GPs to learn the unknown com-

ponents of a control-affine system and form a reachability-based safety metric for reinforcement learning, and [38] utilizes Hamilton-Jacobi reachability to iteratively compute the safe set of a system towards the same end.

Additionally, there exists a body of literature concerning chance-constrained optimization that addresses similar problems. For example, [55] introduces a probabilistic resolvability condition and develops a joint chance-constrained model predictive controller that guarantees this condition for robust control of systems with unbounded stochasticity, and similarly, [56] presents a convex chance-constrained model predictive controller for polynomial, discrete-time stochastic systems. Meanwhile, [57] extends chance-constrained control techniques to handle uncertainty in both the system state and constraint parameters for linear discrete-time stochastic systems, and [58] develops a distributionally robust data-enabled predictive control algorithm for unknown stochastic linear time-invariant systems. We note that these works mainly concern the stochastic setting, while in this work we consider the deterministic unknown setting.

In contrast to these works, we consider general nonlinear systems and approximate multiple probability levels of the unknown disturbance input to compute the optimal control action that is both performant and safe. We achieve this by utilizing previously developed computationally efficient and scalable reachability techniques [51] to overapproximate the different quantiles of the reachable set distribution. These quantiles are able to capture asymmetries that arise from propagating the distribution of possible disturbance inputs through the nonlinear dynamics. Finally, we demonstrate that using these overapproximations in a model predictive control scheme leads to increased performance while maintaining safety on a case study of a kinematic bicycle operating on terrain with an unknown friction coefficient.

The rest of this chapter is structured as follows: section 5.2 formally defines the problem, and then the mixed monotonicity property is leveraged to estimate the probability distribution of reachable sets in section 5.3. We insert this estimation into a control algorithm that balances performance and safety in section 5.4 and demonstrate this control algorithm with our case study in section 5.5.

5.2 Problem Setup

We consider the continuous-time, nonlinear system

$$\dot{x} = f(x, u, w) \tag{5.1}$$

with f Lipschitz continuous in its arguments, where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the system input, and $w \in \mathbb{R}^p$ is a deterministic unknown, state-dependent component of the dynamics such that $w_i = g_i(x)$ where g_i is unknown. We denote by $\phi(t, x_0, \pi)$ the resulting closed-loop state trajectory of (Equation 5.1) under control strategy $u = \pi(t, x)$ when $w = g(x)$ and the system is initialized at x_0 at time 0.

We assume that there exists a known subset of the state space $\mathcal{X}_{\text{unsafe}} \subset \mathbb{R}^n$ which must be avoided, and a goal region $\mathcal{X}_{\text{goal}} \subset \mathbb{R}^n$ which we must drive the system into. Thus, we formally define our problem statement as follows.

Problem Statement. *Consider the system (Equation 5.1). Given known goal region $\mathcal{X}_{\text{goal}}$, compute a feedback control strategy $u = \pi(t, x)$ that reaches the goal in minimal timesteps while avoiding known unsafe region $\mathcal{X}_{\text{unsafe}}$.*

We explored a similar problem setting in [51] where we showcased an exploration-exploitation algorithm ensuring safe control for all time until the goal was reached. However, we did not previously consider performance. Thus, in this work, we develop a formulation for overapproximating multiple probability quantiles of the distribution of reachable sets, and show that accounting for this additional information in the optimization problem that solves for the next control action produces actions that are not only more aggressive and reach the goal in fewer timesteps, but that are also safe.

5.3 Overapproximating the Probability Distribution of Reachable Sets

In this section, we provide an overview of how mixed monotonicity enables efficient calculation of probabilistically correct reachable set overapproximations for systems with unknown components. We then introduce the capability of estimating probability distributions for reachable sets via nested bounding functions that hold with known probability.

If there exist known bounding functions $\underline{\gamma}_i(\underline{x}, \bar{x}; \rho)$ and $\bar{\gamma}_i(\underline{x}, \bar{x}; \rho)$, $\underline{\gamma}_i, \bar{\gamma}_i : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$, for all $i \in \{1, \dots, p\}$, parameterized by $\rho \in [0, 1]$, such that

$$\underline{\gamma}_i(\underline{x}, \bar{x}; \rho) \leq g_i(x) \leq \bar{\gamma}_i(\underline{x}, \bar{x}; \rho), x \in [\underline{x}, \bar{x}], \quad (5.2)$$

holds for all $\underline{x}, \bar{x} \in \mathbb{R}^n$, $\underline{x} \preceq \bar{x}$ with probability at least ρ , then these functions can be inserted into the previously described embedding system to produce valid reachable set overapproximations. This produces a new embedding system parameterized by ρ as

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\bar{x}} \end{bmatrix} = e(\underline{x}, u, \bar{x}; \rho) := \begin{bmatrix} \delta(\underline{x}, u, \underline{\gamma}(\underline{x}, \bar{x}; \rho), \bar{x}, \bar{\gamma}(\underline{x}, \bar{x}; \rho)) \\ \delta(\bar{x}, u, \bar{\gamma}(\underline{x}, \bar{x}; \rho), \underline{x}, \underline{\gamma}(\underline{x}, \bar{x}; \rho)) \end{bmatrix}. \quad (5.3)$$

The resulting state of (Equation 5.3) at time T when initialized at some initial condition $\underline{x}_0, \bar{x}_0$ is thus denoted by $\Phi^\rho(T; (\underline{x}_0, \bar{x}_0), u)$. Moving forward, we use $\underline{\gamma}^\rho, \bar{\gamma}^\rho, e^\rho$ as shorthand for the respective equations with parameter ρ . Likewise, we use $\underline{x}^\rho, \bar{x}^\rho$ to denote the resulting states of e^ρ .

A key property of $\underline{\gamma}^\rho, \bar{\gamma}^\rho$ is that, since they bound the unknown disturbance behavior over the entire state space, the probability that these functions are valid bounds directly translates to the probability that the resulting reachable set overapproximations are also valid [22]. In other words, since $\underline{\gamma}^\rho, \bar{\gamma}^\rho$ hold with probability ρ , the resulting reachable set overapproximations defined by $[\underline{x}^\rho, \bar{x}^\rho]$ also hold with probability at least ρ .

The main insight of this work is that, by considering multiple values of ρ , we obtain richer information regarding the potential future behavior of the system. We impose the following mild

assumption on the construction of $\underline{\gamma}^\rho, \bar{\gamma}^\rho$:

Assumption 6. For all $\rho_1, \rho_2 \in [0, 1], \rho_1 < \rho_2$, it holds that

$$\underline{\gamma}^{\rho_2}(\underline{x}, \bar{x}) \preceq \underline{\gamma}^{\rho_1}(\underline{x}, \bar{x}) \preceq \bar{\gamma}^{\rho_1}(\underline{x}, \bar{x}) \preceq \bar{\gamma}^{\rho_2}(\underline{x}, \bar{x}) \quad (5.4)$$

for all $\underline{x}, \bar{x} \in \mathbb{R}^n$ with $\underline{x} \preceq \bar{x}$.

Assumption 6 states that as ρ increases, the bounding functions always expand, and as ρ decreases, the bounding functions always contract.

Remark 1. We have shown in previous work [22] that it is possible to construct bounds that fulfill Assumption 6 by modeling the unknown functions $g_i(x)$ as GPs, though this is not the only method to produce valid bounding functions, as demonstrated in section 5.5.

This leads to our first result.

Proposition 2. Consider the embedding system (Equation 5.3) with initial condition $\underline{x}_0, \bar{x}_0$. For all $\rho_1, \rho_2 \in [0, 1], \rho_1 < \rho_2$,

$$\Phi^{\rho_2}(T; (\underline{x}_0, \bar{x}_0), u) \preceq_{SE} \Phi^{\rho_1}(T; (\underline{x}_0, \bar{x}_0), u) \quad (5.5)$$

holds for all $T \geq 0$.

The proof of the above follows from Assumption 6 and [22, Theorem 2] and is thus omitted.

We denote by ϱ an ordered finite sequence of discrete ρ percentile values chosen for the embedding system (i.e. the (i) th component of ϱ is always less than or equal to the $(i + 1)$ th component). We demonstrate in section 5.5 that having more than one ρ , and thus including a richer approximation of the probability distribution of the unknown components, enables control strategies that increase performance while maintaining safety.

Finally, suppose the Lipschitz constants of the system are known. In that case, it is possible to overapproximate quantiles of the reachable set distribution which hold with some probability

$\rho_o \notin \varrho$ without directly solving for the trajectory Φ^{ρ_o} . This is encapsulated in the following assumption and theoretical result.

Assumption 7. *The system (Equation 5.1) is Lipschitz continuous with respect to x, w with known Lipschitz constants L_x, L_w . Additionally, the bounding functions $\underline{\gamma}^\rho, \bar{\gamma}^\rho$ are Lipschitz continuous with respect to ρ with Lipschitz constant L_ρ .*

Proposition 3. *Consider (Equation 5.1) and $\underline{\gamma}^\rho, \bar{\gamma}^\rho$ fulfilling Assumption 6 and Assumption 7. Given $\rho_o \in [\varrho_i, \varrho_{i+1}]$, and the resulting tight hyperrectangular reachable set overapproximations $[\underline{x}^{\varrho_i}, \bar{x}^{\varrho_i}]$ and $[\underline{x}^{\varrho_{i+1}}, \bar{x}^{\varrho_{i+1}}]$ at time $t = T$ initialized from $[\underline{x}_0, \bar{x}_0]$, the hyperrectangular reachable set overapproximation $[\underline{x}^{\rho_o}, \bar{x}^{\rho_o}]$ where*

$$\underline{x}^{\rho_o} = \max \left\{ \underline{x}^{\varrho_{i+1}}, \underline{x}^{\varrho_i} - \frac{L_w}{L_x} (e^{L_x T} - 1) L_\rho (\rho_o - \varrho_i) \right\} \quad (5.6)$$

$$\bar{x}^{\rho_o} = \min \left\{ \bar{x}^{\varrho_{i+1}}, \bar{x}^{\varrho_i} + \frac{L_w}{L_x} (e^{L_x T} - 1) L_\rho (\rho_o - \varrho_i) \right\} \quad (5.7)$$

overapproximates the true reachable set of (Equation 5.1) at time $t = T$ with probability at least ρ_o .

Proof. Consider $(\underline{y}^{\rho_o}, \bar{y}^{\rho_o}) = \Phi^{\rho_o}(T; (\underline{x}_0, \bar{x}_0), u)$ with δ tight. The hyperrectangle defined by $[\underline{y}^{\rho_o}, \bar{y}^{\rho_o}]$ overapproximates the true reachable set of (Equation 5.1) with probability at least ρ_o via the properties of $\underline{\gamma}^\rho, \bar{\gamma}^\rho$.

We then consider the two possible values of \bar{x}^{ρ_o} . In the case where $\bar{x}^{\rho_o} = \bar{x}^{\varrho_{i+1}}$, it holds that $\bar{x}^{\rho_o} \succeq \bar{y}^{\rho_o}$ via Proposition 2 as $\varrho_{i+1} > \rho_o$. In the case where $\bar{x}^{\rho_o} = \bar{x}^{\varrho_i} + \frac{L_w}{L_x} (e^{L_x T} - 1) L_\rho (\rho_o - \varrho_i)$, it holds that $\bar{x}^{\rho_o} \succeq \bar{y}^{\rho_o}$ via [59, Corollary 3.17]. Thus, it always holds that $\bar{x}^{\rho_o} \succeq \bar{y}^{\rho_o}$. Similar logic can be used to show that $\underline{x}^{\rho_o} \preceq \underline{y}^{\rho_o}$ always holds as well.

Consequently, since $[\underline{y}^{\rho_o}, \bar{y}^{\rho_o}]$ overapproximates the true reachable set of (Equation 5.1) with probability at least ρ_o , it must hold that $[\underline{x}^{\rho_o}, \bar{x}^{\rho_o}]$ overapproximates the true reachable set of (Equation 5.1) with probability at least ρ_o . ■

5.4 A Control Algorithm to Balance Performance and Safety

We now insert the previously developed approximation of the probability distribution of the system's reachable sets into an optimization problem that, when solved, produces a control strategy that enables high performance while maintaining safety. This optimization is then utilized in a model predictive control framework to produce an algorithm that enables safe, high-performance actions for all time.

We discretize both the original system and the crafted embedding system via forward Euler and then design the following optimization problem:

$$\underset{\Pi=\{u[0],\dots,u[D-1]\}}{\text{minimize}} \quad J([\underline{x}^\rho[0], \bar{x}^\rho[0]], \dots, [\underline{x}^\rho[D], \bar{x}^\rho[D]]) \quad (5.8)$$

subject to:

$$\text{(Equation 5.3), } [\underline{x}^\rho[0], \bar{x}^\rho[0]] = [\underline{x}[0], \bar{x}[0]] \text{ given, } u[d] \in \mathcal{U}$$

$$\forall \rho \in \varrho, \forall d \in \{0, \dots, D-1\}$$

where the cost function is defined as

$$\begin{aligned} J([\underline{x}^\rho[0], \bar{x}^\rho[0]], \dots, [\underline{x}^\rho[D], \bar{x}^\rho[D]]) := & \quad (5.9) \\ & \sum_{\rho \in \varrho} \sum_{d=0}^d \left(a \cdot \text{dist}([\underline{x}^\rho[d], \bar{x}^\rho[d]], \mathcal{X}_{\text{goal}}) \right. \\ & \quad \left. + b \cdot \rho \cdot \text{inter}([\underline{x}^\rho[d], \bar{x}^\rho[d]], \mathcal{X}_{\text{goal}}) \right. \\ & \quad \left. + c \cdot \rho \cdot \text{inter}([\underline{x}^\rho[d], \bar{x}^\rho[d]], \mathcal{X}_{\text{unsafe}}) \right) \end{aligned}$$

where $\text{dist}([\underline{x}^\rho[d], \bar{x}^\rho[d]], \mathcal{X}_{\text{goal}})$ is the distance between the center of each reachable set overapproximation and the center of $\mathcal{X}_{\text{goal}}$, and $\text{inter}([\underline{x}^\rho[d], \bar{x}^\rho[d]], \mathcal{X}_{\text{goal}})$ and $\text{inter}([\underline{x}^\rho[d], \bar{x}^\rho[d]], \mathcal{X}_{\text{unsafe}})$ are the area of intersection between each overapproximation and $\mathcal{X}_{\text{goal}}$ and $\mathcal{X}_{\text{unsafe}}$.

The control strategy Π that solves (Equation 5.8) is such that the overlap with the goal region

is maximized while the overlap with the obstacle is minimized, with a bias toward driving closer to the goal region if the system is too far from it to intersect. The constants a, b, c can be tuned to prioritize performance or safety as needed. In subsection 5.5.1, we demonstrate that a richer approximation (i.e. including multiple values of ρ in ϱ) of the full probability distribution is what enables synthesized control strategies to be both performant and safe.

We then insert the optimization problem (Equation 5.8) into a model predictive control scheme, where the problem is solved to synthesize a control strategy Π , the first control action is executed, and then the optimization problem is run again at the resulting state. This process repeats until the system arrives in the goal region.

We demonstrate in subsection 5.5.2 that, again, including overapproximations of multiple quantiles of the probability distribution produces control actions that are simultaneously aggressive toward the goal and safe from the obstacle. This results in the system taking fewer timesteps to reach the goal while maintaining safety, showcasing that this controller solves our problem statement.

5.5 Case Studies

In this section we present results showcasing the benefit of considering a set of probabilistic reachability distributions compared to exclusively enforcing safety or exclusively assuming best-case. We provide both a Monte-Carlo simulation of a one-shot control scenario as well as a demonstration of the model predictive control formulation¹.

We consider the four-dimensional kinematic planar bicycle, which has state $x = [X, Y, \psi, v]^T$ and relates positional coordinates X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(u_2)$, and front and rear distances from center of mass $l_f = 2.2\text{m}$ and $l_r = 3.3\text{m}$ as

$$\begin{aligned} \dot{X} &= v \cos(\psi + \beta(u_2)), & \dot{Y} &= v \sin(\psi + \beta(u_2)), \\ \dot{\psi} &= \frac{v}{l_r} \sin(\beta(u_2)), & \dot{v} &= u_1, \end{aligned} \tag{5.10}$$

¹Code for both case studies is available at https://github.com/gtfactslab/Cao_ACC2025.

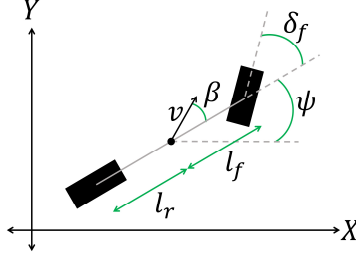


Figure 5.1: The kinematic bicycle model has positions X and Y , center-of-mass velocity v , heading angle ψ , side-slip angle $\beta(\delta_f)$, front and rear distances from center of mass l_f and l_r , and inputs acceleration u_1 and steering angle $u_2 = \delta_f$.

where

$$\beta(u_2) = \arctan\left(\frac{l_r}{l_f + l_r} \tan(u_2)\right), \quad (5.11)$$

with inputs being the desired acceleration u_1 and steering angle u_2 . The bicycle is subject to constraints $\mathcal{U} = [(-10, -1.5), (10, 1.5)]$ and is also affected by an unknown friction coefficient between the wheels and the surface of the road. As a result, the actual velocity update dynamics are given by $\dot{v}_{\text{actual}} = (1 - \mu)u_1$, where $\mu \in [0, 1]$. This system is visualized in Figure 5.1.

The associated decomposition function takes the form

$$\delta(x, u, w, \hat{x}, \hat{w}) = \begin{bmatrix} d^X & d^Y & d^\psi & d^v \end{bmatrix}^T \quad (5.12)$$

$$d^v = d^{b_1 b_2} \left(\begin{bmatrix} 1 - \hat{w} \\ u_1 \end{bmatrix}, \begin{bmatrix} 1 - w \\ u_1 \end{bmatrix} \right)$$

where, for $b, \hat{b} \in \mathbb{R}^2$,

$$d^{b_1 b_2}(b, \hat{b}) = \begin{cases} \min\{b_1 b_2, \hat{b}_1 b_2, b_1 \hat{b}_2, \hat{b}_1 b_2\}, & \text{if } b \preceq \hat{b} \\ \max\{b_1 b_2, \hat{b}_1 b_2, b_1 \hat{b}_2, \hat{b}_1 b_2\}, & \text{if } \hat{b} \preceq b, \end{cases}$$

and d^X, d^Y, d^ψ take the same forms as in [22, Equation 73]. We then construct the associated

embedding system parameterized by ρ as described by (Equation 5.3).

The distribution M which dictates the possible value of μ is known and follows the probability density function

$$f_M(\mu) = \begin{cases} 1.25 & \mu \in [0.2, 0.6) \\ 5 & \mu \in [0.1, 0.2) \\ 0 & \text{otherwise.} \end{cases} \quad (5.13)$$

which was chosen to highlight the effect of nonlinearities on the reachable sets. The bounding functions are

$$\underline{\gamma}^{\{0.25, 0.5, 0.75, 1.0\}}(\underline{x}, \bar{x}) = \{0.175, 0.15, 0.125, 0.1\} \quad (5.14)$$

$$\bar{\gamma}^{\{0.25, 0.5, 0.75, 1.0\}}(\underline{x}, \bar{x}) = \{0.3, 0.4, 0.5, 0.6\}. \quad (5.15)$$

The system is tasked with reaching a goal region defined by $X \in [3.9, 4.9]$, $Y \in [-0.5, 0.5]$ while also avoiding the obstacle next to it defined by $X \in [5, 6]$, $Y \in [-5, 5]$. We apply two different control synthesis strategies and compare their performance when given full distribution information ($\varrho = [0.25, 0.5, 0.75, 1.0]$), when only given a conservative estimation of the worst-case friction values ($\varrho = [1.0]$), and when given an overly confident underestimation of the possible friction values ($\varrho = [0.25]$). We refer to these as Full, Conservative, and Reckless, respectively.

5.5.1 One-Shot Monte Carlo

We first perform a Monte Carlo simulation to illustrate the ability of the Full strategy to increase performance while maintaining safety.

The system is randomly initialized within the hyperrectangle

$$[\underline{x}_0, \bar{x}_0] = [(0.5, -2.51, 0, 0), (0.51, -2.5, 0, 0)] \quad (5.16)$$

and is tasked with synthesizing a one-shot input strategy that will drive the system into the goal

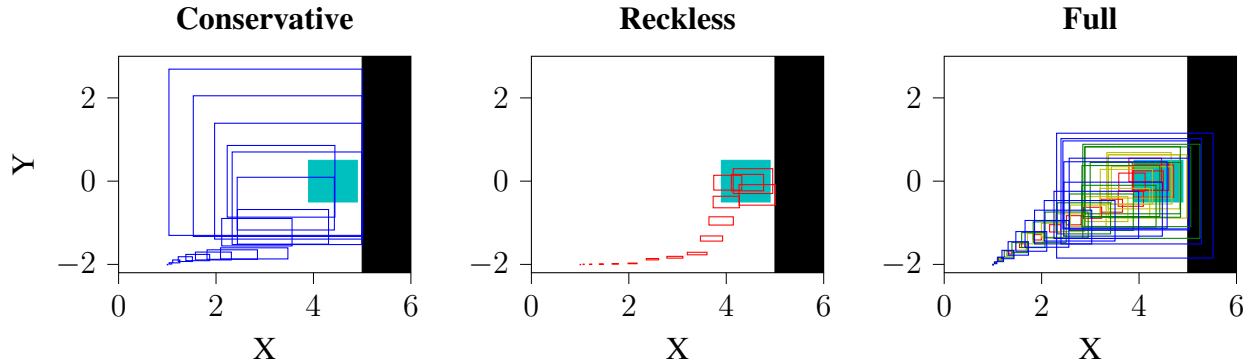


Figure 5.2: Resulting control behaviors in the Monte Carlo simulation, showcasing the estimated reachable sets for each synthesized control strategy from solving (Equation 5.8). With only one level of information, the Conservative and Reckless strategies avoid all overlap with the obstacle. In the Conservative case, this results in a loss of performance, while in the Reckless case, this results in a loss of safety. By contrast, the Full strategy allows for some overlap between the worst-case hyperrectangles and the obstacle, preserving safety while still achieving high performance.

while avoiding the obstacle. Specifically, the system is discretized using timesteps of 0.1 seconds and the one-shot control strategy will be run for $t_f = 1.5$ seconds, resulting in an overall control strategy $\Pi = \{u[0], \dots, u[D - 1]\}$, $D = 15$ obtained by solving (Equation 5.8) with constants $a = 0.005$, $b = -10$, and $c = 10$.

The optimization problem is implemented in Python on a personal laptop computer utilizing the `immrnx` library [60], and solved using IPOPT [61]. After a solution is found for Π , 1000 Monte Carlo iterations are run by randomly determining an initial state within $[\underline{x}_0, \bar{x}_0]$ as well as drawing a value of μ from the known distribution. We then apply the synthesized strategy and count the number of instances out of the 1000 iterations where the system reaches the goal as well as the number of instances where it intersects the obstacle. We allow the full strategy to execute in an open-loop fashion, and thus it is possible for the system to hit both the goal and obstacle in the same iteration, or hit neither. We report these results, as well as the computation time to solve each optimization problem, in Table 5.1.

By considering the full distribution of possible values of μ , the solver finds a control strategy that results in a higher percentage of goals reached without decreasing the safety of the system. Additionally, in Figure 5.2 we showcase the differences in behavior between each level of infor-

Table 5.1: Results from 1000 runs in the Monte Carlo simulation, where the number of runs in which each strategy hit the goal and/or hit the obstacle are recorded. The Full strategy performs better than the Conservative strategy while maintaining the same safety level. While the Reckless strategy achieves the most goal hits, it is also the only strategy that hits the obstacle. We also note that the Full strategy, despite considering four levels of probability, takes less than four times the amount of computation time compared to the other strategies.

Case	Hit Goal	Hit Obstacle	Comp. Time
Conservative	580	0	162 sec.
Reckless	719	256	154 sec.
Full	651	0	527 sec.

mation. With only one level of information, the solver does not allow any intersection between the reachable set overapproximations and the obstacle for the Conservative and Reckless strategies. By contrast, the Full strategy captures the nonlinear effects on the disturbance behavior, and allows the outermost reachable set overapproximations to intersect the obstacle as these represent the worst case scenario.

5.5.2 Model Predictive Control

We now implement the model predictive scheme outlined in section 5.4 to demonstrate that the Full strategy is able to make aggressive but safe maneuvers.

The system is initialized at $x_0 = [1.0, 2.0, 0.0, 0.0]$ and must arrive at the same goal region while avoiding the obstacle defined as the region $X \in [5, 6]$, $Y \in [-10, 10]$. We instantiate ten simulation instances with differing values of μ drawn from the same probability distribution, and record the average computation time per step, the number of timesteps needed to reach the goal, and the total amount of acceleration input applied (i.e. the sum of all applied u_1). The resulting average values of each are listed in Table 5.2.

Overall, the Full strategy arrives at the goal in the fewest timesteps, as it is able to make large moves toward the goal while maintaining correct predictions as to avoid erratic behavior. This is especially apparent in cases like the one illustrated in Figure 5.3, where the value of $\mu = 0.485$ is outside the bounds utilized by the Reckless strategy. This causes the resulting states from the

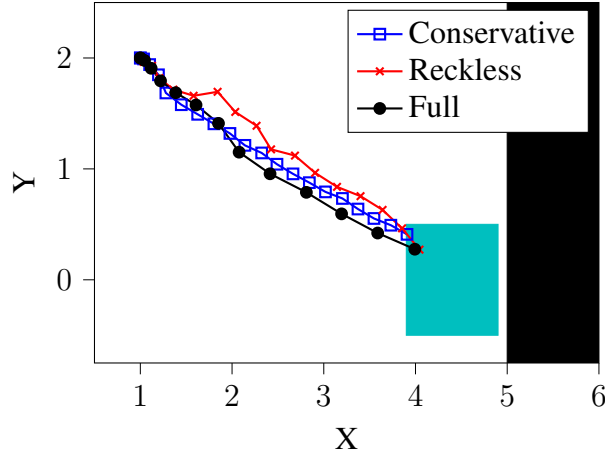


Figure 5.3: Instantiation of the MPC Case Study, where the value of μ is outside the bounds that the Reckless strategy utilizes, causing its reachable set overapproximations to be incorrect and producing erratic behavior. The Conservative strategy has a smooth path, but takes longer than it needs to due to the amount of overapproximation. The Full strategy performs large movements akin to the Reckless strategy and maintains the correct predictions of the Conservative strategy.

Table 5.2: Average step computation time, number of timesteps to goal, and total acceleration applied from 10 runs in the MPC simulation. As shown, the Full strategy arrives at the goal in the fewest number of timesteps and is able to apply the largest amount of acceleration, while not being significantly more computationally expensive.

Case	Timesteps	Total Acceleration	Step Comp. Time
Conservative	16.2	40.15	11.6 sec.
Reckless	13.9	57.38	21.0 sec.
Full	12.3	63.02	75.3 sec.

controller applied by the Reckless strategy to fall outside its predicted reachable sets, causing the system to miss its desired state and producing said erratic behavior. The Conservative bounds encapsulate μ , but result in smaller movements due to the size of the overapproximations. Thus, the Full strategy can make larger movements like the Reckless strategy but still encapsulates the true system behavior like the Conservative strategy.

CHAPTER 6

TRAJECTORY TRACKING RUNTIME ASSURANCE FOR SYSTEMS WITH PARTIALLY UNKNOWN DYNAMICS

6.1 Introduction

Safety-critical autonomous systems often require guarantees that they will only operate within an allowable safe region of their statespace. As a motivating example considered throughout this chapter, in the context of urban air mobility, one key safety requirement is the ability of aerial vehicles to land without damaging the vehicle or the environment around it. As these vehicles are operating in uncontrolled outdoor environments, they are subject to environmental disturbances that are not known a priori. While techniques exist for planning trajectories that are nominally safe [62], ensuring that the system is able to track these trajectories during runtime in the presence of unknown disturbances is challenging. The goal of this work is to design a runtime-assurance framework that ensures safety in the presence of unknown disturbances. Our approach is to compute forward invariant tubes for the system given the limits of the control input and the current knowledge of the disturbance behavior, and override a nominal tracking control law if the system is at risk of violating its safety restrictions.

In many cases, uncertainty in the dynamics can be learned via observations. For example, the position-dependent wind field that an aerial vehicle encounters might be unknown a priori but can be learned through observations of the wind at different points. For such systems, as in [23, 22], we derive high probability bounds on the unknown disturbance behavior by modeling the disturbance as a state-dependent Gaussian Process (GP). In turn, these bounds enable us to calculate overapproximations of the reachable sets of the system that hold with high probability. For this, we again leverage the mixed monotonicity property of dynamical systems to embed the dynamics in a higher dimensional system [1, 5]. The key advantage of this overapproximation technique

that we display in this chapter is its computational efficiency: since it reduces the reachable set computation to the evaluation of a single trajectory of an embedding system with twice the number of states as the original system, this computation is efficient enough to be updated in real-time, even for systems of moderately high dimension [3]. We leverage this property to develop an approach that dynamically detects when a new observation of the disturbance behavior needs to be collected to keep the system within an acceptable bound of the reference trajectory, and then update the reference trajectory and forward invariant tube with the new observation during runtime to preserve safety.

There are several methods for ensuring safety at runtime, which is commonly referred to as runtime assurance. The most common runtime assurance architecture is the Simplex architecture proposed by [63], where two controllers are developed for the system: one that is high-assurance and another that is high-performance. The high-performance controller is allowed to run until some predetermined decision logic detects that the system is about to violate a safety specification, at which point control is switched to the high-assurance controller. This allows the high-performance controller to be developed without having to validate that it is safe beforehand. Alternatively, [64] introduces online active set invariance filtering, which instead minimally modifies the control action such that there exists a back-up trajectory that takes the system to a safe set, while never actually needing to execute the backup strategy. Another approach is introduced in [65], which develops a formal language to implement runtime assurance. An illustrative example of practical runtime assurance for unmanned aerial systems is implemented in [66], though that method is implemented at the waypoint and trajectory selection level.

With regards to assurance for systems with partially unknown dynamics, [67] develops a Twin Neural Lyapunov Function which is then used to build a runtime monitor. However, like many neural network applications, a large amount of training data is needed to ensure safety, and the formulation assumes no prior knowledge of the system's dynamics. Alternatively, [68] develops a runtime assurance mechanism for distributed avionics architecture which can intervene in the event of failure. [69] develops a nested control strategy consisting of an outer task-space loop

and an inner joint-space loop for the trajectory tracking problem on a manipulator with uncertain kinematics and dynamics.

Other approaches to the invariant tube synthesis problem (also known as the *funnel synthesis* problem) include [70], which develops a funnel synthesis algorithm for computing controlled invariant sets around a given nominal trajectory by solving a differential LMI, [71] which develops several strategies utilizing Sum-of-Squares programming for computing regions of finite-time invariance around solutions of polynomial differential equations, and [72] which proposes an approach to funnel synthesis that is based on falsification. Additionally, [73] develops a joint trajectory and funnel synthesis technique for discrete-time systems with locally Lipschitz nonlinearities. Finally, [74, 75] develop funnel synthesis strategies applied in real-time on aerial vehicles.

In this chapter, we leverage the mixed monotonicity property to derive a runtime assurance mechanism that can detect whether the controller is unable to follow the reference trajectory within a desired tolerance. Specifically, we apply the formulation proposed by [76] to calculate a forward-invariant tube around the reference trajectory based on the current observations of the unknown disturbance behavior and the limits of the controller, and we then design a controller which guarantees this forward invariance. Next, we define a safe deviation threshold which, if crossed, triggers an observation of the unknown dynamics and a recalculation of the reference trajectory and forward invariant tube, ensuring that the system never deviates from the reference trajectory by more than the safe threshold. We demonstrate this formulation on a case study of a planar multirotor vehicle making a safe landing in a wind field.

In contrast to the current literature, our proposed formulation natively accommodates nonlinear dynamics without the need to solve for LMIs, which are relatively expensive computationally. Additionally, we solve the problem fully in continuous-time and do not need to choose time discretization points. Finally, we propose a method for sampling to improve knowledge of the unknown dynamics that minimizes the number of samples needed to ensure safety.

The rest of this chapter is structured as follows: We define our problem in section 6.2, and derive our forward-invariant tube formulation and runtime assurance mechanism in section 6.3.

We then apply it to a case study of a planar multirotor vehicle in section 6.4.

6.2 Problem Setup

We consider the continuous-time, nonlinear, Lipschitz-continuous system

$$\dot{x} = f(x, u, w) \quad (6.1)$$

where $x \in \mathbb{R}^n$ is the system state, $u \in U = [\underline{u}, \bar{u}] \subset \mathbb{R}^m$ is the system input constrained to an interval, and $w \in \mathbb{R}^p$ is an unknown, state-dependent component of the dynamics so that $w_i = g_i(x)$ where g_i is unknown.

For this chapter only, we make the following assumptions on (Equation 6.1).

Assumption 8. *The Isaacs minimax condition is satisfied: given any interval sets $U = [\underline{u}, \bar{u}] \subset \mathbb{R}^m$ and $W = [\underline{w}, \bar{w}] \subset \mathbb{R}^p$, for all $q \in \mathbb{R}^n$,*

$$\min_{u \in U} \max_{w \in W} \langle q, f(x, u, w) \rangle = \max_{w \in W} \min_{u \in U} \langle q, f(x, u, w) \rangle. \quad (6.2)$$

This is a mild assumption requiring that, for obtaining an optimal control strategy, it does not matter whether the input is chosen before or after the disturbance is realized at each time instant.

Assumption 9. *For all $(x, w) \in \mathbb{R}^n \times \mathbb{R}^p$, $f(x, U, w)$ is an interval set.*

This assumption is more restrictive, but as we will showcase in section 6.4, it is sometimes possible to apply a transformation to the state-space dynamics of a system such that the transformed dynamics fulfill this assumption. Additionally, when the assumption does not hold, it is always possible to under-approximate $f(x, U, w)$ with interval sets.

Thus, we define our problem as follows.

Problem 5. *Given a system (Equation 6.1) which fulfills Assumption 8 and Assumption 9, as well as reference trajectories $x_r(t), \dot{x}_r(t), u_r(t)$, which solve (Equation 6.1) for some $w_r(t)$, devise an*

assurance mechanism that ensures the system remains within a safe distance ε of the reference trajectory for all time.

Multiple formulations exist to produce safe trajectories for, *e.g.*, autonomous aerial vehicles, leveraging techniques from optimal control [62, 77], and differential flatness [78, 79], for example. In this work, we are specifically interested in the problem of ensuring close tracking of a given reference trajectory in the presence of unknown disturbance behavior at runtime. Thus, we presume that any of these techniques are readily available for generating the reference trajectory.

6.3 High Probability Forward Invariance

In this section, we modify the mixed monotonicity formulation from the calculation of high probability reachable sets to the calculation of a high probability forward invariant tube with respect to a reference trajectory in service of solving Problem 5.

6.3.1 Mixed Monotonicity

We preserve Assumption 2 and adjust the requirements on the decomposition function δ to apply mixed monotonicity on the system (Equation 6.1). The new requirements are as follows:

1. For all $x, u, w, \bar{x}, \bar{u}, \bar{w}$, $\delta(x, u, w, \bar{x}, \bar{u}, \bar{w}) = f(x, u, w)$;
2. For all $\underline{x}, \bar{x}, u, \hat{u}, w, \hat{w}$ and all $i, j \in \{1, \dots, n\}, i \neq j$, $\frac{\partial \delta_i}{\partial \underline{x}_j}(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) \geq 0$;
3. For all $\underline{x}, \bar{x}, u, \hat{u}, w, \hat{w}$ and all $i, j \in \{1, \dots, n\}$, $\frac{\partial \delta_i}{\partial \bar{x}_j}(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) \leq 0$;
4. For all $i \in \{1, \dots, n\}$ and all $k \in \{1, \dots, p\}$, $\frac{\partial \delta_i}{\partial w_k}(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) \geq 0$ and $\frac{\partial \delta_i}{\partial \hat{w}_k}(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) \leq 0$ for all $\underline{x}, \bar{x}, u, \hat{u}, w, \hat{w}$;
5. For all $i \in \{1, \dots, n\}$ and all $k \in \{1, \dots, m\}$, $\frac{\partial \delta_i}{\partial u_k}(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) \geq 0$ and $\frac{\partial \delta_i}{\partial \hat{u}_k}(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) \leq 0$ for all $\underline{x}, \bar{x}, u, \hat{u}, w, \hat{w}$.

6.3.2 Forward Invariant Tube

We now apply the formulation proposed in [76] to generate our forward invariant tube. For any system, a valid decomposition function takes the form

$$\delta_i(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) = \min_{a \in [\underline{x}, \bar{x}], a_i = \underline{x}_i} \min_{b \in [u, \hat{u}]} \min_{c \in [w, \hat{w}]} f_i(a, b, c) \quad (6.3)$$

$$\delta_i(\bar{x}, \hat{u}, \hat{w}, \underline{x}, u, w) = \max_{a \in [\underline{x}, \bar{x}], a_i = \underline{x}_i} \max_{b \in [u, \hat{u}]} \max_{c \in [w, \hat{w}]} f_i(a, b, c) \quad (6.4)$$

for all $i \in [1, n]$. However, solving for these values at runtime is generally difficult; thus the main challenge lies in deriving a closed-form version of δ that solves (Equation 6.3)-(Equation 6.4). A function δ that has this property is considered to be *tight*.

For the particular problem setting of tracking a reference trajectory, the control input u and disturbance input w are strictly competitive forces. In other words, any deviation from $w_r(t)$ taken by the disturbance input w must be countered with deviation from $u_r(t)$ by the control input u . To model this competition, we note that a function δ that is tight (i.e. fulfills (Equation 6.3)–(Equation 6.4)) also has the properties

$$\delta_i(\underline{x}, \hat{u}, w, \bar{x}, u, \hat{w}) = \min_{a \in [\underline{x}, \bar{x}], a_i = \underline{x}_i} \max_{b \in [u, \hat{u}]} \min_{c \in [w, \hat{w}]} f_i(a, b, c) \quad (6.5)$$

$$\delta_i(\bar{x}, u, \hat{w}, \underline{x}, \hat{u}, w) = \max_{a \in [\underline{x}, \bar{x}], a_i = \underline{x}_i} \min_{b \in [u, \hat{u}]} \max_{c \in [w, \hat{w}]} f_i(a, b, c) \quad (6.6)$$

for all $i \in [1, n]$.

Since our application is essentially interested in the worst-case scenario, we can use \underline{u} and \bar{u} moving forward, as these represent the full capability of our controller. Finally, we recall that the reference trajectory $x_r(t)$ solves (Equation 6.1) for some $u_r(t)$ and $w_r(t)$. Such a trajectory can be attained by defining $w_r(t) = \mu_t(x_r(t))$, which lies within the high-probability bounds attained

previously. Thus, as is proposed in [76], we define the embedding system dynamics as

$$\dot{\underline{x}}_i = \begin{cases} d_i(\underline{x}, \bar{x}), & x_i(t) < x_{ri}(t) \\ \min\{d_i(\underline{x}, \bar{x}), \dot{x}_{ri}(t)\}, & x_i(t) \geq x_{ri}(t) \end{cases} \quad (6.7)$$

$$\dot{\bar{x}}_i = \begin{cases} d_i(\bar{x}, \underline{x}), & x_i(t) > x_{ri}(t) \\ \max\{d_i(\bar{x}, \underline{x}), \dot{x}_{ri}(t)\}, & x_i(t) \leq x_{ri}(t) \end{cases} \quad (6.8)$$

where

$$d_i(\underline{x}, \bar{x}) = \delta_i(\underline{x}, \bar{u}, \underline{\gamma}(\underline{x}, \bar{x}), \bar{x}, \underline{u}, \bar{\gamma}(\underline{x}, \bar{x})), \quad (6.9)$$

$$d_i(\bar{x}, \underline{x}) = \delta_i(\bar{x}, \underline{u}, \bar{\gamma}(\underline{x}, \bar{x}), \underline{x}, \bar{u}, \underline{\gamma}(\underline{x}, \bar{x})). \quad (6.10)$$

The resulting hyperrectangular tube formed by $[\underline{x}(t), \bar{x}(t)]$ is a controlled invariant tube, that is, a tube that the system is able to remain within given the limits of the controller. Again, as we are using bounds on the disturbance that hold with probability at least $1 - \eta$, this property holds with the same probability.

We now characterize a class of safety assurance control policies. At runtime, given current state $x(t)$, any control strategy that satisfies

$$u(x(t)) \in \underset{u \in [\underline{u}, \bar{u}]}{\operatorname{argmin}} \max_{w \in [\underline{\gamma}, \bar{\gamma}]} \langle p(x(t)), f(x(t), u, w) \rangle \quad (6.11)$$

where

$$p_i(x(t)) = \begin{cases} 1, & x_i(t) \geq \bar{x}_i(t) \\ -1, & x_i(t) \leq \underline{x}_i(t) \\ 0, & \text{otherwise} \end{cases}$$

guarantees forward invariance of $[\underline{x}(t), \bar{x}(t)]$ with probability at least $1 - \eta$.

Moreover, if at some time it does not hold that

$$\bigwedge_{i=1}^n |\underline{x}_i(t) - x_{ri}(t)| \leq \varepsilon \wedge |\bar{x}_i(t) - x_{ri}(t)| \leq \varepsilon \quad (6.12)$$

for all $t \geq 0$, then an observation of the unknown disturbance behavior $g(x)$ is triggered at the time t_r where

$$t_r = \min_{t \geq 0} |\underline{x}_i(t) - x_{ri}(t)| \geq \varepsilon \vee |\bar{x}_i(t) - x_{ri}(t)| \geq \varepsilon, \quad (6.13)$$

after which a new reference trajectory is generated and the forward invariant tube is recalculated.

The overall runtime assurance mechanism is outlined in algorithm 4.

Algorithm 4 Runtime Assurance Mechanism

Data: Embedding system (Equation 6.7)-(Equation 6.8), bounding functions $\underline{\gamma}, \bar{\gamma}$, safety threshold

ε
 $t_r \leftarrow 0$;

while *In Operation* **do**

if $t \geq t_r$ **then**

 Generate Reference Trajectories x_r, \dot{x}_r, u_r which solve (Equation 6.1) for $w_r \in [\underline{\gamma}, \bar{\gamma}]$,

$x_r(t) = x(t)$;

$\underline{x}, \bar{x} \leftarrow$ solutions to (Equation 6.7), (Equation 6.8), $\underline{x}(t) = \bar{x}(t) = x(t)$;

$t_r \leftarrow$ (Equation 6.13);

 Apply u satisfying (Equation 6.11);

This brings us to the main theoretical result of this chapter.

Theorem 10. *Given reference trajectories $x_r(t), \dot{x}_r(t), u_r(t)$ that solve Equation 6.1 for $w_r(t) = \mu_t(x_r(t))$, applying the runtime assurance mechanism outlined by algorithm 4 guarantees that no state in the system deviates from its reference trajectory by more than ε for all $t > 0$.*

Proof. At the instant $t = t_r$, the reference trajectory is calculated with $x_r(t) = x(t)$ and the tube is initialized with $\underline{x}(t) = \bar{x}(t) = x(t)$. Thus, it must always hold that the next $t_r > t$ given $\varepsilon > 0$ and condition (Equation 6.13). Consequently, we can say that $t \leq t_r$ for all $t > 0$ at runtime.

We then note that, per condition (Equation 6.13), it must hold that

$$[\underline{x}(t), \bar{x}(t)] \subseteq [x_r(t) - \varepsilon, x_r(t) + \varepsilon] \forall t \leq t_r. \quad (6.14)$$

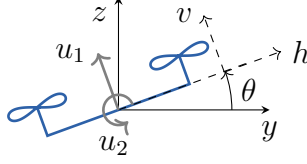


Figure 6.1: The planar multirotor model has horizontal position y , vertical position z , and roll angle θ . The inputs are thrust u_1 in the direction perpendicular to the line segment connecting the rotors and roll angle acceleration u_2 . The modified five dimensional state x consists of y , z , θ , and the velocity in the horizontal and vertical directions of the frame of the multirotor h and v , so that $x = [y \ h \ z \ v \ \theta]^T$, which fulfills Assumption 9.

Since $t \leq t_r$ for all $t > 0$ at runtime, then for all time $t > 0$,

$$[\underline{x}(t), \bar{x}(t)] \subseteq [x_r(t) - \varepsilon, x_r(t) + \varepsilon]. \quad (6.15)$$

Finally, per [76, Theorem 1], the hyperrectangular tube defined by $[\underline{x}(t), \bar{x}(t)]$ is forward invariant given a control strategy that satisfies (Equation 6.11). ■

Thus, the outlined formulation solves Problem 5.

6.4 Aerial Vehicle Case Study

We now apply the forward invariant tube and runtime assurance mechanism to an example of a multirotor system that is constrained to moving within the vertical plane. The five-dimensional state x of the planar multirotor system consists of horizontal position y , vertical position z , roll angle θ , and the derivatives \dot{y} and \dot{z} , so that $x = [y \ \dot{y} \ z \ \dot{z} \ \theta]^T$. The two inputs are thrust u_1 acting at the center of mass in the direction $\begin{bmatrix} -\sin \theta & \cos \theta \end{bmatrix}^T$ perpendicular to the line segment connecting the rotors, and roll angular velocity u_2 .

We assume the system is subject to input constraints $[\underline{u}, \bar{u}] = [(-5, -5), (15, 5)]$ and gravitational acceleration $a_g = 9.81m/s^2$, as well as an unknown force due to wind. We assume this force affects acceleration in the horizontal direction and is a function of altitude z . The resulting

dynamics with normalized mass and moment of inertia are

$$\begin{aligned}
\ddot{y} &= -u_1 \sin \theta + g(z) \\
\ddot{z} &= u_1 \cos \theta - a_g \\
\dot{\theta} &= u_2
\end{aligned} \tag{6.16}$$

where $g(z)$ constitutes the unknown wind force in the horizontal direction.

We first note that the dynamics as outlined in (Equation 6.16) do not fulfill Assumption 9. Thus, we introduce state variables v and h to denote the vertical and horizontal velocity of the quadcopter *in its own frame*. In effect, this is applying a rotation based on θ to the original velocity dynamics \dot{y} and \dot{z} . The transformed state-space dynamics take the form

$$\begin{aligned}
\dot{y} &= h \cos \theta - v \sin \theta \\
\dot{h} &= -a_g \sin \theta + g(z) \cos \theta \\
\dot{z} &= h \sin \theta + v \cos \theta \\
\dot{v} &= u_1 - a_g \cos \theta - g(z) \sin \theta \\
\dot{\theta} &= u_2
\end{aligned} \tag{6.17}$$

which now fulfill Assumption 9. These dynamics are illustrated in Figure 6.1. The resulting decomposition function is

$$\delta(\underline{x}, u, w, \bar{x}, \hat{u}, \hat{w}) = \left[d^y \quad d^h \quad d^z \quad d^v \quad u_2 \right]^T \tag{6.18}$$

with

$$d^y = d^{b_1 b_2} \left(\begin{bmatrix} \underline{h} \\ d^{\cos}(\underline{\theta}, \bar{\theta}) \end{bmatrix}, \begin{bmatrix} \bar{h} \\ d^{\cos}(\bar{\theta}, \underline{\theta}) \end{bmatrix} \right) - d^{b_1 b_2} \left(\begin{bmatrix} \bar{v} \\ d^{\sin}(\bar{\theta}, \underline{\theta}) \end{bmatrix}, \begin{bmatrix} \underline{v} \\ d^{\sin}(\underline{\theta}, \bar{\theta}) \end{bmatrix} \right)$$

$$\begin{aligned}
d^h &= -a_g d^{\sin}(\bar{\theta}, \underline{\theta}) + d^{b_1 b_2} \left(\begin{bmatrix} w \\ d^{\cos}(\underline{\theta}, \bar{\theta}) \end{bmatrix}, \begin{bmatrix} \hat{w} \\ d^{\cos}(\bar{\theta}, \underline{\theta}) \end{bmatrix} \right) \\
d^z &= d^{b_1 b_2} \left(\begin{bmatrix} \underline{h} \\ d^{\sin}(\underline{\theta}, \bar{\theta}) \end{bmatrix}, \begin{bmatrix} \bar{h} \\ d^{\sin}(\bar{\theta}, \underline{\theta}) \end{bmatrix} \right) + d^{b_1 b_2} \left(\begin{bmatrix} \underline{v} \\ d^{\cos}(\underline{\theta}, \bar{\theta}) \end{bmatrix}, \begin{bmatrix} \bar{v} \\ d^{\cos}(\bar{\theta}, \underline{\theta}) \end{bmatrix} \right) \\
d^v &= u_1 - a_g d^{\cos}(\bar{\theta}, \underline{\theta}) - d^{b_1 b_2} \left(\begin{bmatrix} \hat{w} \\ d^{\sin}(\bar{\theta}, \underline{\theta}) \end{bmatrix}, \begin{bmatrix} w \\ d^{\sin}(\underline{\theta}, \bar{\theta}) \end{bmatrix} \right)
\end{aligned}$$

where $d^{b_1 b_2}$, d^{\sin} , and d^{\cos} are as in [22, Equations 74-75].

To generate a reference trajectory, we linearize the system (Equation 6.17) around the equilibrium and then employ a Linear-Quadratic Regulator (LQR) feedback controller with parameters $Q = \text{diag}([1000, 500, 20, 500, 1])$ and $R = \text{diag}([20, 20])$ to simulate a trajectory to the origin assuming g behaves according to the current mean of the Gaussian process estimation of the disturbance. The resulting state and input trajectories form the reference trajectories x_r, \dot{x}_r, u_r that we use in calculating the forward invariant tube.

We then designate a safe landing hyperrectangle $\mathcal{X}_{\text{land}} := [(-0.5, -0.1, -1, -0.1, -\pi/6), (-0.5, 0.1, 0, 0.1, \pi/6)]$ and then calculate the forward invariant tube of the system using the formulation outlined in section 6.3 and the current estimated bounds on g .

We calculate the applied control action as $u(x(t)) = [u_1(v(t)), u_2(\theta(t))]^T$ where, for $i = \{1, 2\}$,

$$u_i(x(t)) = u_{ri}(t) + \begin{cases} \left(\frac{x(t) - x_r(t)}{\bar{x}(t) - x_r(t)} \right)^3 (u_{ri}(t) - \underline{u}_i), & x(t) \geq x_r(t) \\ \left(\frac{x_r(t) - x(t)}{x_r(t) - \underline{x}(t)} \right)^3 (\bar{u}_i - u_{ri}(t)), & x(t) \leq x_r(t) \end{cases} \quad (6.19)$$

which fulfills the condition (Equation 6.11). If there does not exist a time t such that $[\underline{x}(t), \bar{x}(t)] \subseteq \mathcal{X}_{\text{land}}$, then we collect a new observation of the disturbance when any edge of the tube deviates from the reference trajectory more than the safety threshold $\varepsilon = 0.3$, and then recalculate the reference trajectory and forward invariant tube. The simulation of the system was executed in Simulink on a personal computer, the results of which are outlined in Figure 6.2.

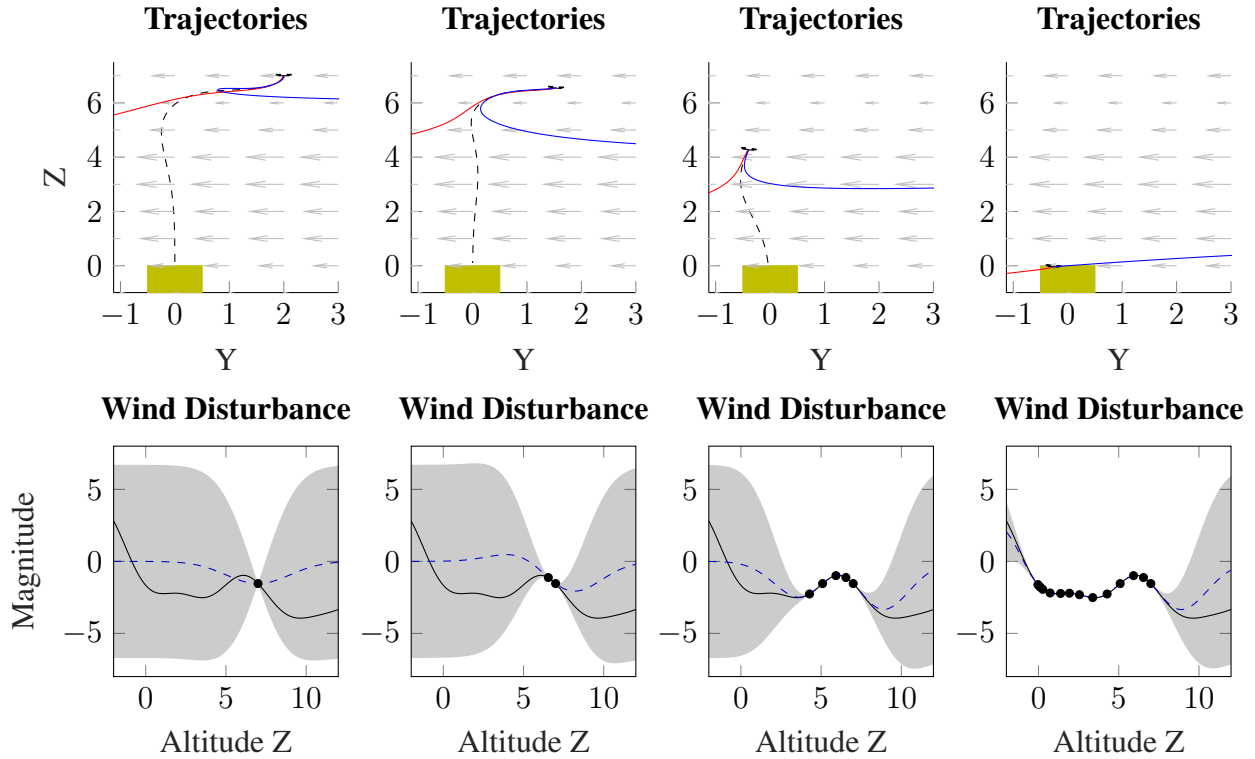


Figure 6.2: The planar multirotor landing simulation. The multirotor attempts to make a safe landing by following the calculated reference trajectory (top, black dashed), which is a solution to (Equation 6.1) assuming the wind behaves according to the current estimated mean (bottom, blue dashed) based on the available observations (bottom, points) of the true wind behavior (bottom, black, and top, arrows). We then derive the forward invariant tube (top, red and blue) around the reference trajectory, which assumes the worst-case wind behavior, calculated by the current confidence bounds (bottom, shaded) on said behavior. At the point in the trajectory where the invariant tube deviates from the reference by a certain threshold, a recalculation of the reference trajectory and invariant tube is triggered. This process repeats until the multirotor makes a safe landing.

As shown, the quadcopter is initially unable to guarantee a safe landing; the forward invariant tube quickly expands past the safety threshold, as it must account for the uncertainty of the disturbance behavior when the reference trajectory enters regions with few observations of the disturbance. As the quadcopter descends, collecting observations and updating the reference trajectory and forward invariant tube, it is eventually able to achieve a safe landing. The calculation of the reference trajectories and forward invariant tube takes between 0.08–0.2 seconds, and a recalculation is triggered every 0.75–2.0 seconds, showcasing the formulation’s real-time capabilities.

CHAPTER 7

TRAJECTORY TRACKING FOR SYSTEMS WITH UNKNOWN TIME-VARYING DISTURBANCES

7.1 Introduction

We build upon our work in the previous chapter, which focused on developing a runtime assurance mechanism for trajectory tracking for systems with unknown time-invariant disturbance behavior. We expand upon this work by introducing a mixed Jacobian method for constructing a general embedding system that produces a high-probability forward invariant tube around the reference trajectory without an interval assumption on the dynamics, and incorporating time-varying disturbance behaviors into the formulation.

7.2 Problem Setup

We consider the continuous-time, nonlinear, Lipschitz-continuous system

$$\dot{x} = f(x, u, w) \tag{7.1}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the system input, and $w \in \mathbb{R}^p$ is an unknown, time-varying, state-dependent component of the dynamics so that $w_i = g_i(t, x)$ where g_i is unknown.

Given a set of reference trajectories, our objective is to efficiently compute a forward invariant tube around said reference trajectories that holds with high probability. Thus, we formally define our problem as follows.

Problem 6. *Given a system (Equation 7.1) and reference trajectories x_r, u_r , produce a controlled forward-invariant tube around the reference trajectory that holds with probability at least $1 - \eta$.*

Multiple formulations exist to produce safe trajectories for, *e.g.*, autonomous aerial vehicles,

leveraging techniques from optimal control [62, 77], and differential flatness [78, 79], for example. In this work, we are specifically interested in the problem of calculating a forward invariant tube around the given reference trajectory in the presence of unknown disturbance behavior at runtime. Thus, we presume that any of these techniques are readily available for generating the reference trajectory.

7.3 Time-Varying Gaussian Process Bounds

We have shown previously in [22] that for time-invariant disturbance functions $g_i(x)$, modeling these functions as GPs enables the formulation of bounding functions that fulfill the time-invariant version of (Equation 2.3) with probability $1 - \eta, \eta \in (0, 1]$. We now extend these results to the time-varying setting and derive bounds that fulfill (Equation 2.3) with probability $1 - \eta$.

We pull from [80] and model g using a GP with a spatiotemporal kernel model with temporal kernel of the form

$$(1 - \epsilon)^{\frac{|t_1 - t_2|}{2}} \quad (7.2)$$

where $\epsilon \in [0, 1]$ represents the rate of change in g over time. $\epsilon = 1$ recovers the time-invariant model of g , while $\epsilon = 0$ means that the behavior of g is entirely independent between any two observation times.

We first review the standard time-invariant model. Given a set of τ observations $\{y_j\}_{j=1}^\tau$ of the GP at corresponding points $\{x_j\}_{j=1}^\tau$, the surrogate functions of interest to approximate g_i are

$$\forall i \in \{1, \dots, p\}, \quad \begin{cases} \bar{g}_i^{(\tau)}(x) := \mu_\tau(x) + \sqrt{\beta_\tau} \sigma_\tau(x) \\ \underline{g}_i^{(\tau)}(x) := \mu_\tau(x) - \sqrt{\beta_\tau} \sigma_\tau(x) \end{cases} \quad (7.3)$$

where β_τ is chosen as per [22, Theorem 7], $\mu_\tau(\cdot)$ is the posterior mean, and $\sigma_\tau(\cdot)$ is the posterior variance, computed according to the standard GP updates [9]:

$$\mu_\tau(x) := k_\tau(x)^T (K_\tau + \sigma^2 I)^{-1} y_\tau \quad (7.4)$$

$$k_\tau(x, x') := k(x, x') - k_\tau(x)^T (K_\tau + \sigma^2 I)^{-1} k_\tau(x') \quad (7.5)$$

$$\sigma_\tau^2(x) := k_\tau(x, x) \quad (7.6)$$

where $k_\tau(x) := (k(x_1, x), \dots, k(x_\tau, x))$ and $K_\tau = [k_\tau(x_i, x_j)]$. To extend this formulation into the time-varying case, we now leverage that the observations $\{y_j\}_{j=1}^\tau$ are taken at times $\{t_j\}_{j=1}^\tau$. Given temporal kernel (Equation 7.2), the updated posterior mean and variance equations for querying a point at time $t_{\tau+1}$ are

$$\tilde{\mu}_{\tau+1}(x) := \tilde{k}_\tau(x)^T (\tilde{K}_\tau + \sigma^2 I)^{-1} y \quad (7.7)$$

$$k_{\tau+1}(x, x') := k(x, x') - \tilde{k}_\tau(x)^T (\tilde{K}_\tau + \sigma^2 I)^{-1} \tilde{k}_\tau(x') \quad (7.8)$$

$$\tilde{\sigma}_{\tau+1}^2(x) := k_{\tau+1}(x, x) \quad (7.9)$$

with

$$\tilde{K}_\tau = K_\tau \odot [(1 - \epsilon)^{|t_i - t_j|/2}]_{i,j=1}^\tau \quad (7.10)$$

$$\tilde{k}_\tau(x) = k_\tau(x) \odot [(1 - \epsilon)^{(t_{\tau+1} - t_i)/2}]_{i=1}^\tau \quad (7.11)$$

where \odot is the Hadamard product. Thus, the surrogate functions of interest to approximate time-varying g_i are

$$\forall i \in \{1, \dots, p\}, \quad \begin{cases} \bar{g}_i^{(\tau+1)}(x) := \tilde{\mu}_{\tau+1}(x) + \sqrt{\beta_\tau} \tilde{\sigma}_{\tau+1}(x) \\ \underline{g}_i^{(\tau+1)}(x) := \tilde{\mu}_{\tau+1}(x) - \sqrt{\beta_\tau} \tilde{\sigma}_{\tau+1}(x) \end{cases} \quad (7.12)$$

Then, for all $i \in \{1, \dots, p\}$ and all $\tau \geq 1$, define for all $\underline{x} \preceq \bar{x}$,

$$\underline{\gamma}_i(\tau + 1, \underline{x}, \bar{x}) := \min_{x \in [\underline{x}^{(\tau, -)}, \bar{x}^{(\tau, +)}] \cap \mathcal{D}_\tau} \underline{g}_i^{(\tau+1)}(x) - \frac{1}{(\tau + 1)^2}, \quad (7.13)$$

$$\bar{\gamma}_i(\tau + 1, \underline{x}, \bar{x}) := \max_{x \in [\underline{x}^{(\tau, -)}, \bar{x}^{(\tau, +)}] \cap \mathcal{D}_\tau} \bar{g}_i^{(\tau+1)}(x) + \frac{1}{(\tau + 1)^2}. \quad (7.14)$$

where $\underline{x}^{(\tau,-)}$, $\bar{x}^{(\tau,+)}$ are defined as in [22, Equations 54-55]. These functions fulfill (Equation 2.3) with probability at least $1 - \eta$, and thus the reachable set overapproximations calculated by the embedding system (Equation 2.7) using these functions hold with probability at least $1 - \eta$.

7.4 Feedback Aware Forward Invariance

In this section, we develop a Jacobian-based general embedding system formulation for calculating a forward invariant tube around a reference trajectory. This new formulation incorporates a known feedback control strategy, allowing for the tube to be controlled forward invariant and thus enabling the runtime assurance mechanism to work on general nonlinear systems of the form (Equation 7.1).

We first introduce the mixed Jacobian operator, which is defined as follows. Given some $x \in \mathbb{R}_a$ and a differentiable function $f : \mathbb{R}^a \rightarrow \mathbb{R}^b$, define the mixed Jacobian operator $M_{x'}$ such that $M_{x'} f : \mathbb{R}^a \times [0, 1]^a \rightarrow \mathbb{R}^{a \times b}$ where

$$(M_{x'} f(x, s))_{ij} = \frac{\partial f_i}{\partial x_j}(x_1, \dots, x_{j-1}, s_j x_j + (1 - s_j) x'_j, x'_{j+1}, \dots, x'_a) \quad (7.15)$$

for all $i = 1, \dots, b$ and $j = 1, \dots, a$. Given an interval set $X = X_1 \times \dots \times X_b$ and some $x' \in X$, we define an interval matrix $[\mathcal{M}]$ that overapproximates the mixed Jacobian of (Equation 7.1) over X . In other words, $[\mathcal{M}]$ is such that

$$(M_{x'} f(x, s))_{ij} \subseteq [\mathcal{M}]_{ij} \quad (7.16)$$

for all $i = 1, \dots, b$ and $j = 1, \dots, a$. Per [81, Corollary 1], this is fulfilled if

$$\frac{\partial f_i}{\partial x_j}(X_1, \dots, X_j, x'_{j+1}, \dots, x'_a) \subseteq [\mathcal{M}]_{ij} \quad (7.17)$$

for all $i = 1, \dots, b$ and $j = 1, \dots, a$. As a result, it must hold that

$$f(x) - f(x') \in [\mathcal{M}](x - x'). \quad (7.18)$$

We expand (Equation 7.16) to fit systems of the form (Equation 7.1). First, we define the interval sets $X = X_1 \times \dots \times X_n$, $U = U_1 \times \dots \times U_m$, $W = W_1 \times \dots \times W_p$ as well as some $x' \in X$, $u' \in U$, and $w' \in W$. Thus, (Equation 7.16) becomes

$$(M_{(x',u',w')}f(x, u, w, s))_{ij} \subseteq \left[\mathcal{M}_x \quad \mathcal{M}_u \quad \mathcal{M}_w \right]_{ij} \quad (7.19)$$

for all $i = 1, \dots, n$ and $j = 1, \dots, n + m + p$ over X, U, W . Applying (Equation 7.17) results in this requirement being fulfilled if

$$\frac{\partial f_i}{\partial x_j}(X_1, \dots, X_j, x'_{j+1}, \dots, x'_n, u'_1, \dots, u'_m, w'_1, \dots, w'_p) \subseteq [\mathcal{M}_x]_{ij} \quad (7.20)$$

for all $i, j = 1, \dots, n$,

$$\frac{\partial f_i}{\partial u_j}(X_1, \dots, X_n, U_1, \dots, U_j, u'_{j+1}, \dots, u'_m, w'_1, \dots, w'_p) \subseteq [\mathcal{M}_u]_{ij} \quad (7.21)$$

for all $i = 1, \dots, n$, $j = 1, \dots, m$, and

$$\frac{\partial f_i}{\partial w_j}(X_1, \dots, X_n, U_1, \dots, U_m, W_1, \dots, W_j, w'_{j+1}, \dots, w'_p) \subseteq [\mathcal{M}_w]_{ij} \quad (7.22)$$

for all $i = 1, \dots, n$, $j = 1, \dots, p$. It consequently holds that

$$f(x, u, w) - f(x', u, w) \in [\mathcal{M}_x](x - x'), \quad (7.23)$$

$$f(x, u, w) - f(x, u', w) \in [\mathcal{M}_u](u - u'), \quad (7.24)$$

$$f(x, u, w) - f(x, u, w') \in [\mathcal{M}_w](w - w'). \quad (7.25)$$

Thus, combining (Equation 7.23), (Equation 7.24), and (Equation 7.25) gives

$$f(x, u, w) - f(x', u', w') \in [\mathcal{M}_x](x - x') + [\mathcal{M}_u](u - u') + [\mathcal{M}_w](w - w'). \quad (7.26)$$

As a result, given interval sets $X := [x, \hat{x}]$, $U := [\underline{u}, \bar{u}]$, $W := [\underline{w}, \bar{w}]$, and reference trajectories x_r, u_r, w_r initialized within X, U, W , we formulate generalized embedding system dynamics for (Equation 7.1) as

$$\begin{aligned} [\dot{x}, \hat{\dot{x}}] := & [\mathcal{M}_x]([x, \hat{x}] - x_r) + [\mathcal{M}_u]([\underline{u}, \bar{u}] - u_r) \\ & + [\mathcal{M}_w]([\underline{w}, \bar{w}] - w_r) + f(x_r, u_r, w_r). \end{aligned} \quad (7.27)$$

Next, since we are attempting to track a reference trajectory, we incorporate a known feedback control strategy into our forward invariant tube calculation. We consider a feedback controller of the form

$$u = K(x - x_r) + u_r \quad (7.28)$$

which, when inserted into the generalized embedding system, results in dynamics

$$[\dot{x}, \hat{\dot{x}}] := ([\mathcal{M}_x] + K[\mathcal{M}_u])([x, \hat{x}] - x_r) + [\mathcal{M}_w]([\underline{w}, \bar{w}] - w_r) + f(x_r, u_r, w_r). \quad (7.29)$$

The above embedding system produces a valid forward invariant tube around the reference trajectory, and the previously derived γ bounds and mean behavior $\mu(x)$ can be inserted to produce a tube that holds with high probability.

We further refine our forward invariant tube calculation by leveraging the fact that the disturbance is state-dependent, as it may be possible for the disturbance behavior to have a stabilizing effect on the reference trajectory tracking in some areas of the state space. To capture this, we define the disturbance reference trajectory as $w_r = g(t, x)$ and recall our bounding functions $\underline{\gamma}, \bar{\gamma}$ that fulfill (Equation 2.3). Inserting these bounds into (Equation 7.29) gives

$$\begin{aligned} [\dot{x}, \hat{\dot{x}}] := & ([\mathcal{M}_x] + K[\mathcal{M}_u])([x, \hat{x}] - x_r) \\ & + [\mathcal{M}_w]([\underline{\gamma}(t, x, \hat{x}), \bar{\gamma}(t, x, \hat{x})] - g(t, x_r)) \\ & + f(x_r, u_r, g(t, x_r)). \end{aligned} \quad (7.30)$$

We note that these bounds are such that, for all $i = 1, \dots, p$,

$$\underline{\gamma}_i(t, \underline{x}, \bar{x}) = \min_{x \in [\underline{x}, \bar{x}]} g_i(t, x) - d_i(t, x) \quad (7.31)$$

$$\bar{\gamma}_i(t, \underline{x}, \bar{x}) = \max_{x \in [\underline{x}, \bar{x}]} g_i(t, x) + d_i(t, x) \quad (7.32)$$

where $d(t, x) \in \mathbb{R}_{\geq 0}^p$ represents the difference between $\underline{\gamma}, \bar{\gamma}$ and the true disturbance bounds. We then note that

$$\underline{\gamma}(t, \underline{x}, \bar{x}) \geq \min_{x \in [\underline{x}, \bar{x}]} g(t, x) - \max_{x \in [\underline{x}, \bar{x}]} d(t, x) \quad (7.33)$$

$$\bar{\gamma}(t, \underline{x}, \bar{x}) \leq \max_{x \in [\underline{x}, \bar{x}]} g(t, x) + \max_{x \in [\underline{x}, \bar{x}]} d(t, x) \quad (7.34)$$

must therefore hold. Thus, it must also hold that

$$\begin{aligned} & [\underline{\gamma}(t, x, \hat{x}), \bar{\gamma}(t, x, \hat{x})] - g(t, x_r) \\ & \subseteq \left[\min_{x \in [x, \hat{x}]} g(t, x) - \max_{x \in [x, \hat{x}]} d(t, x), \right. \\ & \quad \left. \max_{x \in [x, \hat{x}]} g(t, x) + \max_{x \in [x, \hat{x}]} d(t, x) \right] \\ & \quad - g(t, x_r). \end{aligned} \quad (7.35)$$

We define $d_{\text{MAX}}(t, \underline{x}, \bar{x}) = \max_{x \in [\underline{x}, \bar{x}]} d(t, x)$ and rearrange terms on the right side to get

$$\begin{aligned} & [\underline{\gamma}(t, x, \hat{x}), \bar{\gamma}(t, x, \hat{x})] - g(t, x_r) \\ & \subseteq \left[\min_{x \in [x, \hat{x}]} g(t, x), \max_{x \in [x, \hat{x}]} g(t, x) \right] - g(t, x_r) \\ & \quad + [-1, 1] \times d_{\text{MAX}}(t, x, \hat{x}) \end{aligned} \quad (7.36)$$

We then apply the mixed Jacobian operator to the disturbance function g and obtain an interval matrix $[\mathcal{M}_x^g]$ that fulfills (Equation 7.16) for $g(t, x)$. It therefore holds that, for any interval $X =$

$X_1 \times \dots \times X_n$ and some $x' \in X$,

$$g(t, x) - g(t, x') \in [\mathcal{M}_x^g](x - x') \quad (7.37)$$

for all $x \in X$. Thus, for $X := [x, \hat{x}]$, it must hold that

$$[\min_{x \in [x, \hat{x}]} g(t, x), \max_{x \in [x, \hat{x}]} g(t, x)] - g(t, x_r) \subseteq [\mathcal{M}_x^g]([x, \hat{x}] - x'). \quad (7.38)$$

Inserting this into (Equation 7.36) gives

$$[\underline{\gamma}(t, x, \hat{x}), \bar{\gamma}(t, x, \hat{x})] - g(t, x_r) \subseteq [\mathcal{M}_x^g]([x, \hat{x}] - x_r) + [-1, 1] \times d_{\text{MAX}}(t, x, \hat{x}) \quad (7.39)$$

which, when inserted into (Equation 7.30), results in embedding system dynamics

$$\begin{aligned} [\dot{x}, \hat{x}] := & ([\mathcal{M}_x] + K[\mathcal{M}_u] + [\mathcal{M}_w][\mathcal{M}_x^g])([x, \hat{x}] - x_r) \\ & + [\mathcal{M}_w]([-1, 1] \times d_{\text{MAX}}(t, x, \hat{x})) \\ & + f(x_r, u_r, g(t, x_r)). \end{aligned} \quad (7.40)$$

Finally, as the actual $g(t, x)$ is unknown, by proxy we define $g(t, x) := \tilde{\mu}_t(x)$ and $d(t, x) := \tilde{\sigma}_t(x)$ as defined by the GP update functions (Equation 7.7) and (Equation 7.9) from section 7.3. Thus, we derive the associated $\sigma_{\text{MAX}}(t, \underline{x}, \bar{x}) = \max_{x \in [\underline{x}, \bar{x}]} \tilde{\sigma}_t(x)$ and $[\mathcal{M}_x^\mu]$, resulting in dynamics

$$\begin{aligned} [\dot{x}, \hat{x}] := & ([\mathcal{M}_x] + K[\mathcal{M}_u] + [\mathcal{M}_w][\mathcal{M}_x^\mu])([x, \hat{x}] - x_r) \\ & + [\mathcal{M}_w]([-1, 1] \times \sigma_{\text{MAX}}(t, x, \hat{x})) \\ & + f(x_r, u_r, \tilde{\mu}_t(x_r)). \end{aligned} \quad (7.41)$$

This embedding system formulation captures the first-order interactions between the time-varying disturbance behavior and the system state, which results in a more accurate forward-

invariant tube around the reference trajectory. Thus, it solves Problem 6. In the next sections, we demonstrate a simple numerical example showcasing this property, and then implement our forward invariant tube formulation on several case studies with a planar multirotor system.

7.5 Case Studies

In this section we provide demonstrations of our forward invariant tube formulation in various applications. We first provide a numerical example illustrating the benefit of incorporating the first-order interactions between the disturbance behavior and the system. We then showcase the formulation on a five-dimensional planar multirotor example via a trajectory tracking scenario as well as an implicit active set invariance filtering scenario.

7.5.1 Numerical Example

We begin with a simple numerical example showcasing the effects of incorporating the first-order interactions of the observed disturbance behavior on the system. We consider the dynamics

$$\dot{x} = x + u + w \tag{7.42}$$

where the disturbance behavior $w = g(x)$ is dictated by the function $g(x) = -x$. We assume we have estimated disturbance bounds

$$\underline{\gamma}(t, \underline{x}, \bar{x}) = \min_{x \in [\underline{x}, \bar{x}]} -x - 1 \tag{7.43}$$

$$\bar{\gamma}(t, \underline{x}, \bar{x}) = \max_{x \in [\underline{x}, \bar{x}]} -x + 1 \tag{7.44}$$

and known reference controller behavior $u_r = -x$ as well as feedback controller behavior $u = -(x - x_r) + u_r$. We initialize the system at $x = 5$ and then craft two embedding systems: one without modeling the first-order disturbance interactions (i.e. (Equation 7.30)), and one with them (i.e. (Equation 7.41)). We initialize both embedding systems at $[x, \hat{x}] = [4.75, 5.25]$ and simulate

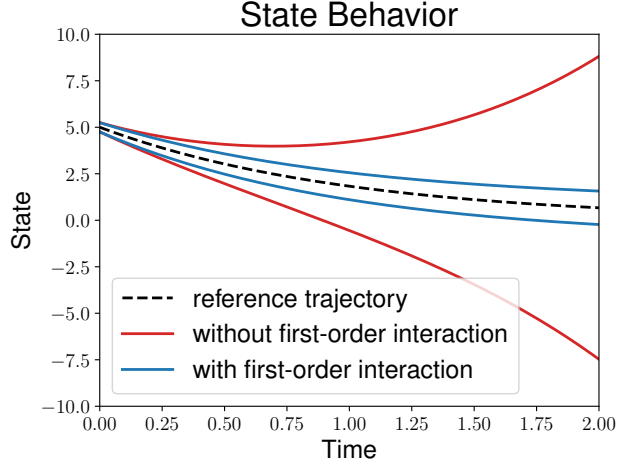


Figure 7.1: Demonstration of the advantages of including first-order interactions of the disturbance in the forward invariant tube calculation. By accounting for these interactions, the tube remains tight around the reference trajectory.

a 2 second trajectory. As shown in Figure 7.1, by accounting for the first-order interactions between the disturbance behavior and the system, the forward invariant tube remains tight around the reference trajectory, rather than quickly expanding.

7.5.2 Trajectory Tracking Runtime Assurance

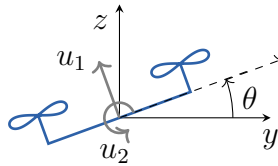


Figure 7.2: The planar multirotor model has horizontal position y , vertical position z , and roll angle θ . The inputs are thrust u_1 in the direction perpendicular to the line segment connecting the rotors and roll angle acceleration u_2 .

We now consider a multirotor system constrained to moving within the vertical plane, and apply our forward invariant tube calculation to the runtime assurance mechanism outlined in algorithm 4. The five-dimensional state x of the planar multirotor system consists of horizontal position y , vertical position z , roll angle θ , and the derivatives \dot{y} and \dot{z} , so that $x = \begin{bmatrix} y & \dot{y} & z & \dot{z} & \theta \end{bmatrix}^T$. The two inputs are thrust u_1 acting at the center of mass in the direction $\begin{bmatrix} -\sin \theta & \cos \theta \end{bmatrix}^T$ perpendicular

to the line segment connecting the rotors, and roll angular velocity u_2 . Additionally, there exists a time and altitude-dependent wind force $g(t, z)$ acting horizontally which is unknown a priori. Thus, the normalized dynamics of the system are

$$\begin{aligned}\ddot{y} &= -u_1 \sin \theta + g(t, z) \\ \ddot{z} &= u_1 \cos \theta - a_g \\ \dot{\theta} &= u_2\end{aligned}\tag{7.45}$$

We then implement the runtime assurance algorithm outlined in algorithm 4 and insert the forward invariant tube calculation (Equation 7.41). We first generate a reference trajectory by linearizing the system (Equation 7.45) around the equilibrium and then employing a Linear-Quadratic Regulator (LQR) feedback controller with parameters $Q = \text{diag}([1000, 500, 20, 500, 1])$ and $R = \text{diag}([20, 20])$ to simulate a trajectory to the origin assuming $g(t, z) = \tilde{\mu}_t(z_r)$. The resulting state and input trajectories form the reference trajectories x_r, u_r that we use to calculate the forward invariant tube.

We then designate a safe landing hyperrectangle

$$\mathcal{X}_{\text{land}} := [(-0.5, -0.1, -1, -0.1, -\pi/6), (-0.5, 0.1, 0, 0.1, \pi/6)]\tag{7.46}$$

and then calculate the forward invariant tube of the system using embedding system (Equation 7.41) and the current estimated $\tilde{\mu}_t, \tilde{\sigma}_t$. We apply the control strategy (Equation 7.28) and calculate the future time t_r at which any state deviates beyond the reference trajectory by ε or more. At that time, we collect an observation of the disturbance behavior and recalculate the reference trajectory and forward invariant tube. This guarantees that the system remains within ε of the reference trajectory for all time. The overall runtime assurance mechanism is outlined in algorithm 5.

As shown in Figure 7.3, the multirotor is initially unable to guarantee a safe landing as the forward invariant tube quickly expands past the allowed threshold. As the quadcopter descends, we collect observations and update the reference trajectory and forward invariant tube, taking into

Algorithm 5 Runtime Assurance Mechanism

Data: Embedding system (Equation 7.41), safety threshold ε , feedback coefficient matrix K

$t_r \leftarrow 0$

while *In Operation* **do**

if $t \geq t_r$ **then**

 Generate Reference Trajectories x_r, u_r which solve (Equation 7.1) for $w_r = \tilde{\mu}_t(x_r)$,

$x_r(t) = x(t)$;

$\underline{x}, \bar{x} \leftarrow$ solution to (Equation 7.41), $\underline{x}(t) = \bar{x}(t) = x(t)$;

$t_r \leftarrow \min_{t \geq 0} |\underline{x}_i(t) - x_{ri}(t)| \geq \varepsilon \vee |\bar{x}_i(t) - x_{ri}(t)| \geq \varepsilon$;

 Apply $u = K(x - x_r) + u_r$;

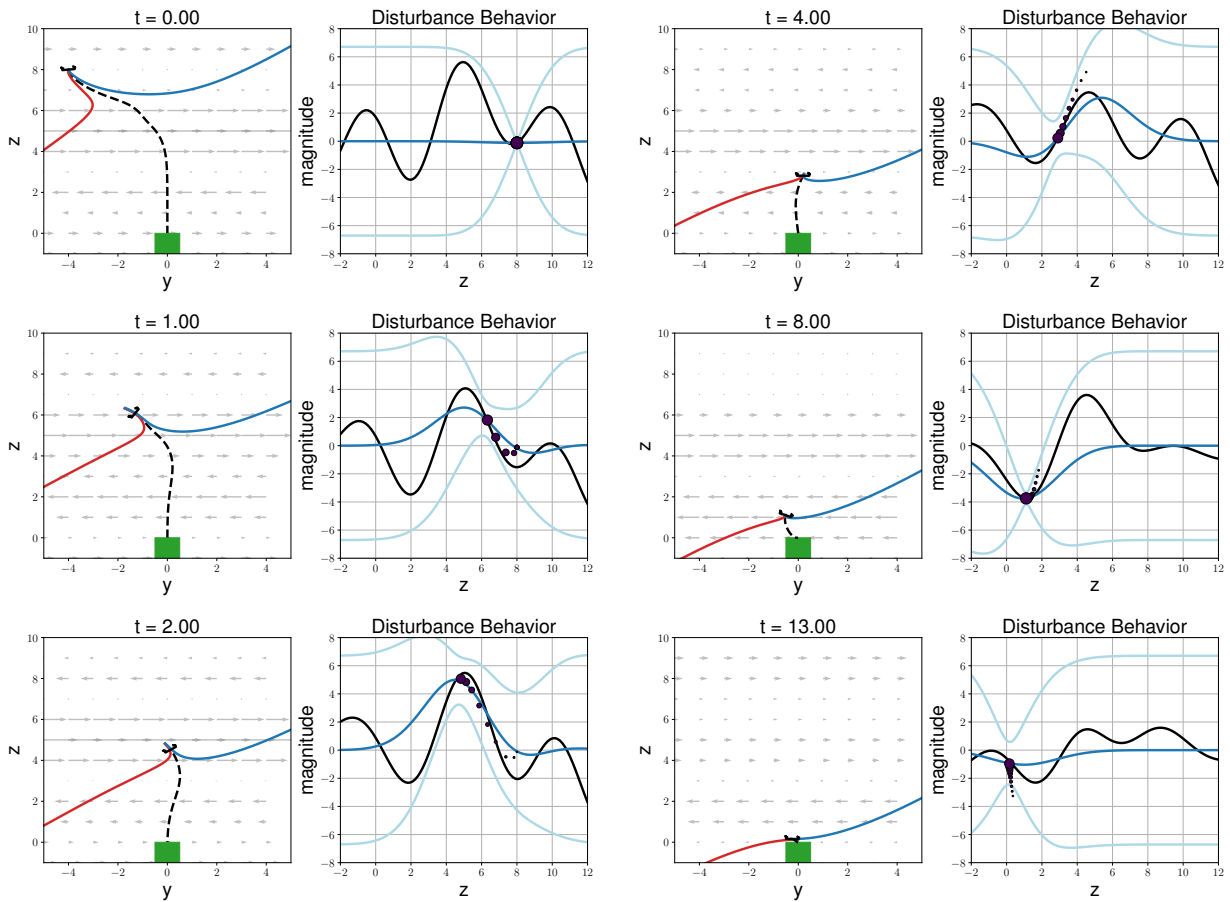


Figure 7.3: The planar multirotor trajectory tracking simulation. The multirotor attempts to track a landing reference trajectory (left, dashed), while the runtime assurance algorithm calculates a forward invariant tube (left, red and blue) using observations (right, dots, largest most recent) to estimate the mean and confidence bounds (right, dark and light blue respectively) on the disturbance (right, solid black). Per algorithm 5, an observation is collected and the trajectories are recomputed whenever the forward invariant tube deviates from the reference beyond the safety threshold.

account the time at which each observation was taken and weighting them appropriately in the confidence bounds on the disturbance behavior. The multirotor is eventually able to achieve a safe landing even in the presence of this time-varying disturbance behavior. We implement this simulation on a personal computer with the `immrnx` library [60]. Calculating the reference trajectories and the forward invariant tube takes approximately 0.04 seconds, and a recalculation is triggered approximately every 0.3-1.0 seconds, showcasing the real-time capabilities of the formulation.

7.5.3 Implicit Active Set Invariance Filtering

We now consider a scenario in which a separate, unverified control strategy is developed for landing, and insert our trajectory tracking formulation into an Implicit Active Set Invariance Filtering formulation to act as a safety filter for said controller. We define a function $h(x)$ and a resulting safety set \mathcal{S} such that

$$\mathcal{S} := \{x \in \mathbb{R}^n | h(x) \geq 0\}, \quad (7.47)$$

which, for this case study, is defined as

$$h(x) := -(10(y^2 - 0.0625) - z) \quad (7.48)$$

which produces the region outlined in green in Figure 7.4.

We then implement an Implicit Active Set Invariance Filter (see, e.g., [82]), which minimally perturbs the desired input such that a backup trajectory that drives the system into \mathcal{S} always exists. Our embedding system formulation (Equation 7.41) produces the backup trajectory and forward invariant tube used in the filter. As shown in Figure 7.4, when there is no guaranteed path back to the safety set (i.e., no slice of the forward invariant tube is fully contained in the safety set), an observation is taken and the filter steps in and drives the system with the maximum force back toward it. This continues until a safe landing is achieved. The filter takes around 0.01 seconds to compute on average, showcasing its capabilities for real-time computation.

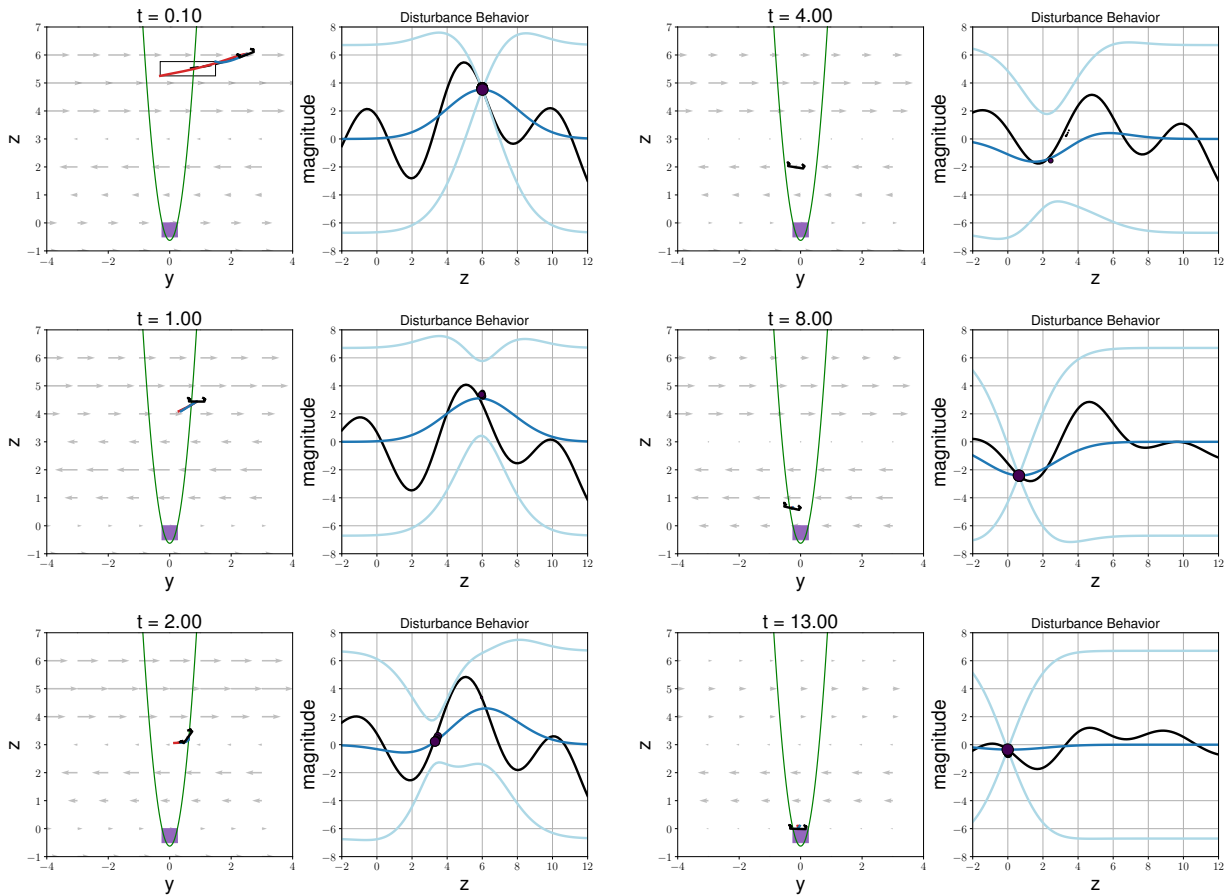


Figure 7.4: The planar multirotor Implicit ASIF simulation. The multirotor attempts to land via an uncertified controller. At all times, the filter calculates the least intrusive modification to the desired input such that a backup trajectory (left, red and blue) back to the safety region (left, green) always exists. The formulation incorporates observations (right, dots, largest most recent) to estimate the mean and confidence bounds (right, dark and light blue respectively) on the disturbance (right, solid black), which are collected whenever no part of the forward invariant tube fully fits within the safety region.

REFERENCES

- [1] S. Coogan, “Mixed monotonicity for reachability and safety in dynamical systems,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 5074–5085.
- [2] S. Coogan, E. A. Gol, M. Arcak, and C. Belta, “Traffic network control from temporal logic specifications,” *IEEE Transactions on Control of Network Systems*, vol. 3, no. 2, pp. 162–172, Jun. 2016.
- [3] M. Abate, M. Mote, E. Feron, and S. Coogan, “Verification and runtime assurance for dynamical systems with uncertainty,” in *Hybrid Systems: Computation and Control*, 2021.
- [4] M. Abate and S. Coogan, “Computing robustly forward invariant sets for mixed-monotone systems,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 4553–4559.
- [5] G. Enciso, H. Smith, and E. Sontag, “Nonmonotone systems decomposable into monotone systems with negative feedback,” *Journal of Differential Equations*, vol. 224, no. 1, pp. 205–227, 2006.
- [6] D. Angeli, G. A. Enciso, and E. D. Sontag, “A small-gain result for orthant-monotone systems under mixed feedback,” *Systems & Control Letters*, vol. 68, pp. 9–19, 2014.
- [7] S. Coogan and M. Arcak, “Efficient finite abstraction of mixed monotone systems,” in *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, 2015, pp. 58–67.
- [8] H. Smith, “Global stability for mixed monotone systems,” *Journal of Difference Equations and Applications*, vol. 14, no. 10-11, pp. 1159–1164, 2008.
- [9] C. E. Rasmussen and C. K. I. Williams., *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts: MIT Press, 2006, ISBN: 026218253X.
- [10] F. Berkenkamp, A. P. Schoellig, and A. Krause, “Safe controller optimization for quadrotors with gaussian processes,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 491–496.
- [11] M. Liu, G. Chowdhary, B. C. da Silva, S.-Y. Liu, and J. P. How, “Gaussian processes for learning and control: A tutorial with examples,” *IEEE Control Systems Magazine*, vol. 38, no. 5, pp. 53–86, Oct. 2018.
- [12] W. Luo, C. Nam, G. Kantor, and K. Sycara, “Distributed environmental modeling and adaptive sampling for multi-robot sensor coverage,” in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS ’19, Montreal

QC, Canada: International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1488–1496, ISBN: 978-1-4503-6309-9.

- [13] A. Benevento, M. Santos, G. Notarstefano, K. Paynabar, M. Bloch, and M. Egerstedt, “Multi-robot coordination for estimation and coverage of unknown spatial fields,” in *Proc. of IEEE International Conference on Robotics and Automation*, Paris, France, May 2020, pp. 7740–7746.
- [14] A. Lederer, J. Umlauft, and S. Hirche, “Uniform error bounds for gaussian process regression with application to safe control,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Vancouver, Canada: Curran Associates, Inc., Dec. 2019.
- [15] M. J. Khojasteh, V. Dhiman, M. Franceschetti, and N. Atanasov, “Probabilistic safety constraints for learned high relative degree system dynamics,” in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, vol. 120, PMLR, Jun. 2020, pp. 781–792.
- [16] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, “Learning-based model predictive control for safe exploration,” in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 6059–6066.
- [17] J. J. Choi, F. Castañeda, C. J. Tomlin, and K. Sreenath, “Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions,” in *Proceedings of Robotics: Science and Systems*, Corvallis, Oregon, USA, Jul. 2020.
- [18] F. Castañeda, J. J. Choi, B. Zhang, C. J. Tomlin, and K. Sreenath, “Gaussian process-based min-norm stabilizing controller for control-affine systems with uncertain input effects and dynamics,” in *2021 American Control Conference (ACC)*, 2021, pp. 3683–3690.
- [19] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, “A general safety framework for learning-based control in uncertain robotic systems,” *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2019.
- [20] S. Herbert, J. J. Choi, S. Sanjeev, M. Gibson, K. Sreenath, and C. J. Tomlin, “Scalable learning of safety guarantees for autonomous systems using hamilton-jacobi reachability,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 5914–5920.
- [21] M. Abate, M. Dutreix, and S. Coogan, “Tight decomposition functions for continuous-time mixed-monotone systems with disturbances,” *IEEE Control Systems Letters*, vol. 5, no. 1, pp. 139–144, 2021.

- [22] M. E. Cao, M. Bloch, and S. Coogan, “Efficient learning of hyperrectangular invariant sets using gaussian processes,” *IEEE Open Journal of Control Systems*, vol. 1, pp. 223–236, 2022.
- [23] M. E. Cao, M. Bloch, and S. Coogan, “Estimating high probability reachable sets using gaussian processes,” in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 3881–3886.
- [24] D. Angeli and E. D. Sontag, “Monotone control systems,” *IEEE Transactions on Automatic Control*, vol. 48, no. 10, pp. 1684–1698, 2003.
- [25] H. Smith, *Monotone Dynamical Systems: An Introduction to the Theory of Competitive and Cooperative Systems*. Mathematical surveys and monographs, American Mathematical Society, 2008.
- [26] F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause, “Safe learning of regions of attraction for uncertain, nonlinear systems with Gaussian processes,” in *Proc. of the IEEE Conference on Decision and Control (CDC)*, 2016, pp. 4661–4666.
- [27] N. Srinivas, A. Krause, S. Kakade, and M. Seeger, “Gaussian process optimization in the bandit setting: No regret and experimental design,” in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010, pp. 1015–1022, ISBN: 978-1-60558-907-7.
- [28] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, “Information-theoretic regret bounds for gaussian process optimization in the bandit setting,” *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 3250–3265, 2012.
- [29] P. Polack, F. Althé, B. Novel, and A. de La Fortelle, “The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?” In *2017 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2017, pp. 812–818.
- [30] I. M. Mitchell and J. A. Templeton, “A toolbox of hamilton-jacobi solvers for analysis of nondeterministic continuous and hybrid systems,” in *Hybrid Systems: Computation and Control*, M. Morari and L. Thiele, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 480–494, ISBN: 978-3-540-31954-2.
- [31] M. Althoff, “An introduction to cora 2015,” in *Proc. of the workshop on applied verification for continuous and hybrid systems*, 2015, pp. 120–151.
- [32] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, “Safe model-based reinforcement learning with stability guarantees,” in *Advances in Neural Information Processing Systems*, I. Guyon *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

- [33] M. A. Khan, T. Ibuki, and A. Chatterjee, “Gaussian control barrier functions: Non-parametric paradigm to safety,” *IEEE Access*, vol. 10, pp. 99 823–99 836, 2022.
- [34] P. Jagtap, G. J. Pappas, and M. Zamani, “Control barrier functions for unknown nonlinear systems using gaussian processes,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 3699–3704.
- [35] V. Dhiman, M. J. Khojasteh, M. Franceschetti, and N. Atanasov, “Control barriers in bayesian learning of system dynamics,” *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 214–229, 2023.
- [36] C. Knuth, G. Chou, N. Ozay, and D. Berenson, “Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5129–5136, 2021.
- [37] M. Bujarbaruah, C. Vallon, and F. Borrelli, “Learning to satisfy unknown constraints in iterative mpc,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 6204–6209.
- [38] J. C. Shih, F. Meier, and A. Rai, “A framework for online updates to safe sets for uncertain dynamics,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5994–6001.
- [39] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, “Reachability-based safe learning with gaussian processes,” in *53rd IEEE Conference on Decision and Control*, 2014, pp. 1424–1431.
- [40] M. Bujarbaruah, S. H. Nair, and F. Borrelli, “A semi-definite programming approach to robust adaptive mpc under state dependent uncertainty,” in *2020 European Control Conference (ECC)*, 2020, pp. 960–965.
- [41] J. Köhler, P. Kötting, R. Soloperto, F. Allgöwer, and M. A. Müller, “A robust adaptive model predictive control framework for nonlinear uncertain systems,” *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8725–8749, 2021. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/rnc.5147>.
- [42] U. Rosolia and F. Borrelli, “Learning model predictive control for iterative tasks. a data-driven control framework,” *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2018.
- [43] M. E. Cao and S. Coogan, “Safe learning-based predictive control from efficient reachability,” in *2023 American Control Conference (ACC)*, 2023, pp. 1832–1837.
- [44] B. Potočnik, G. Mušič, and B. Zupančič, “Model predictive control of discrete-time hybrid systems with discrete inputs,” *ISA Transactions*, vol. 44, no. 2, pp. 199–211, 2005.

- [45] T. Lew, A. Sharma, J. Harrison, A. Bylard, and M. Pavone, “Safe active dynamics learning and control: A sequential exploration–exploitation framework,” *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 2888–2907, 2022.
- [46] J. Manzano, J. Calliess, D. M. de la Pena, and D. Limon, “Online learning robust mpc: An exploration-exploitation approach,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5292–5297, 2020.
- [47] K. P. Wabersich and M. Zeilinger, “Bayesian model predictive control: Efficient model exploration and regret bounds using posterior sampling,” in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, A. M. Bayen *et al.*, Eds., ser. Proceedings of Machine Learning Research, vol. 120, PMLR, Jun. 2020, pp. 455–464.
- [48] L. Sforni, I. Notarnicola, and G. Notarstefano, “Learning-driven nonlinear optimal control via gaussian process regression,” in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 4412–4417.
- [49] M. Buisson-Fenet, F. Solowjow, and S. Trimpe, “Actively learning gaussian process dynamics,” in *Learning for dynamics and control*, PMLR, 2020, pp. 5–15.
- [50] D. Sun, M. J. Khojasteh, S. Shekhar, and C. Fan, “Uncertain-aware safe exploratory planning using gaussian process and neural control contraction metric,” in *Learning for Dynamics and Control*, PMLR, 2021, pp. 728–741.
- [51] M. E. Cao and S. Coogan, “Safe learning-based predictive control from efficient reachability,” in *AACC American Control Conference (ACC)*, 2023.
- [52] S. Pohland, S. Herbert, and C. Tomlin, “Efficient safe learning for robotic systems in unstructured environments,” in *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW)*, 2019, pp. 82–86.
- [53] M. Althoff, O. Stursberg, and M. Buss, “Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization,” in *2008 47th IEEE Conference on Decision and Control*, 2008, pp. 4042–4048.
- [54] M. Althoff, O. Stursberg, and M. Buss, “Model-based probabilistic collision detection in autonomous driving,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 299–310, 2009.
- [55] M. Ono, “Joint chance-constrained model predictive control with probabilistic resolvability,” in *2012 American Control Conference (ACC)*, 2012, pp. 435–441.
- [56] A. Jasour and C. Lagoa, “Convex chance constrained model predictive control,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 6204–6209.

- [57] M. P. Vitus, Z. Zhou, and C. J. Tomlin, “Stochastic control with uncertain parameters via chance constrained control,” *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 2892–2905, 2016.
- [58] J. Coulson, J. Lygeros, and F. Dörfler, “Distributionally robust chance constrained data-enabled predictive control,” *IEEE Transactions on Automatic Control*, vol. 67, no. 7, pp. 3289–3304, 2022.
- [59] F. Bullo, *Contraction Theory for Dynamical Systems*, 1.2. Kindle Direct Publishing, 2024, ISBN: 979-8836646806.
- [60] A. Harapanahalli, S. Jafarpour, and S. Coogan, “ImmraX: A parallelizable and differentiable toolbox for interval analysis and mixed monotone reachability in jax,” *IFAC-PapersOnLine*, vol. 58, no. 11, pp. 75–80, 2024, 8th IFAC Conference on Analysis and Design of Hybrid Systems ADHS 2024.
- [61] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Mathematical programming*, vol. 106, pp. 25–57, 2006.
- [62] H. Heidari and M. Saska, “Trajectory planning of quadrotor systems for various objective functions,” *Robotica*, vol. 39, no. 1, pp. 137–152, 2021.
- [63] L. Sha, “Using simplicity to control complexity,” *IEEE Software*, vol. 18, no. 4, pp. 20–28, 2001.
- [64] 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)*, 2018, pp. 3592–3599.
- [65] I. Lee, S. Kannan, M. Kim, O. Sokolsky, and M. Viswanathan, “Runtime assurance based on formal specifications,” *Departmental Papers (CIS)*, p. 294, 1999.
- [66] J. D. Schierman, M. D. DeVore, N. D. Richards, and M. A. Clark, “Runtime assurance for autonomous aerospace systems,” *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 12, pp. 2205–2217, 2020.
- [67] Z. Xiong, J. Eappen, A. H. Qureshi, and S. Jagannathan, “Model-free neural lyapunov control for safe robot navigation,” in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 5572–5579.
- [68] S. Ghori, T. Khamvilai, E. Feron, and M. Pakmehr, “Runtime assurance for distributed avionics architecture,” in *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)*, 2022, pp. 1–6.

- [69] M. Tarokh, “Manipulator task space trajectory tracking with kinematics and dynamics uncertainties,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 9884–9890.
- [70] T. Kim, P. Elango, T. P. Reynolds, B. Açıkmeşe, and M. Mesbahi, “Optimization-based constrained funnel synthesis for systems with lipschitz nonlinearities via numerical optimal control,” *IEEE Control Systems Letters*, vol. 7, pp. 2875–2880, 2023.
- [71] M. M. Tobenkin, I. R. Manchester, and R. Tedrake, “Invariant funnels around trajectories using sum-of-squares programming,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 9218–9223, 2011, 18th IFAC World Congress.
- [72] J. Fejlek and S. Ratschan, “Computing funnels using numerical optimization based falsifiers,” in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 4318–4324.
- [73] T. Kim, P. Elango, and B. Acikmese, *Joint synthesis of trajectory and controlled invariant funnel for discrete-time systems with locally lipschitz nonlinearities*, 2023. arXiv: 2209.03535 [math.OC].
- [74] T. Reynolds, D. Malyuta, M. Mesbahi, B. Acikmese, and J. M. Carson, “Funnel synthesis for the 6-dof powered descent guidance problem,” in *AIAA Scitech 2021 Forum*. eprint: <https://arc.aiaa.org/doi/pdf/10.2514/6.2021-0504>.
- [75] A. Majumdar and R. Tedrake, “Funnel libraries for real-time robust feedback motion planning,” *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017. eprint: <https://doi.org/10.1177/0278364917712421>.
- [76] V. Sinyakov and A. Girard, “Abstraction of continuous-time systems based on feedback controllers and mixed monotonicity,” *IEEE Transactions on Automatic Control*, pp. 1–15, 2022.
- [77] M. Hehn and R. D’Andrea, “Quadrocopter trajectory generation and control,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 1485–1491, 2011, 18th IFAC World Congress.
- [78] R. M. Murray, M. Rathinam, and W. Sluis, “Differential flatness of mechanical control systems: A catalog of prototype systems,” in *ASME international mechanical engineering congress and exposition*, Citeseer, 1995.
- [79] A. Chamseddine, Y. Zhang, C. A. Rabbath, C. Join, and D. Theilliol, “Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 2832–2848, 2012.
- [80] I. Bogunovic, J. Scarlett, and V. Cevher, “Time-varying gaussian process bandit optimization,” in *Proceedings of the 19th International Conference on Artificial Intelligence and*

Statistics, A. Gretton and C. C. Robert, Eds., ser. Proceedings of Machine Learning Research, vol. 51, Cadiz, Spain: PMLR, May 2016, pp. 314–323.

- [81] A. Harapanahalli and S. Coogan, *A linear differential inclusion for contraction analysis to known trajectories*, 2024. arXiv: 2411.11587 [eess.SY].
- [82] C. Llanes, M. Abate, and S. Coogan, “Safety from fast, in-the-loop reachability with application to uavs,” in *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*, 2022, pp. 127–136.

VITA



Michael Enqi Cao was born on May 24th, 1996, in Buffalo, New York. After moving to Georgia in the early 2000s, he earned a Bachelor’s degree in Computer Engineering with a Minor in Robotics in 2018, as well as a Master’s degree in Electrical and Computer Engineering in 2020 from the Georgia Institute of Technology. He finished his journey at the Georgia Institute of Technology in 2025, after completing a PhD in Robotics under the supervision of his advisor, Dr. Samuel Coogan. Along the way, he has also worked for several research organizations, including the NASA Jet Propulsion Laboratory, Sandia National Laboratories, and the Johns Hopkins University Applied Physics Laboratory, as well as companies such as Qualcomm and Airbus. He is a member of the IEEE Control Systems Society and the IEEE Robotics and Automation Society.