UC Berkeley
UC Berkeley Electronic Theses and Dissertations

Title

Designing Assurance Frameworks for Autonomous Systems

Permalink

https://escholarship.org/uc/item/0gw3v0x0

ISBN

9798173788368

Author

Deglurkar, Sampada

Publication Date

2026-08-10

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Designing Assurance Frameworks for Autonomous Systems

by

Sampada Deglurkar

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Claire J. Tomlin, Chair
Professor Shankar Sastry
Professor Sanjit A. Seshia
Professor Marco Pavone

Summer 2026

Designing Assurance Frameworks for Autonomous Systems

Copyright 2026
by
Sampada Deglurkar

Abstract

Designing Assurance Frameworks for Autonomous Systems

by

Sampada Deglurkar

Doctor of Philosophy in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Claire J. Tomlin, Chair

Autonomous systems that operate in safety-critical domains are in need of *assurance frameworks*; that is, tools and techniques for preventing failures during deployment and verifying that a system will satisfy specifications. Designing assurance frameworks is especially difficult for learning-enabled systems since machine learning models are not easily interpretable, work in high-dimensional spaces, do not have guarantees on their outputs, and often operate in unstructured and complex environments. The central problem of this dissertation is therefore related to the *principles of design* for assurance frameworks: what makes a good assurance framework, and how do its components interact with each other and the overall system? The main challenge is that tools such as uncertainty quantification, conformal prediction, and scenario optimization, while well-studied on their own and promising for producing assurances, are not well-understood in the context of safety and control.

In the first part of this dissertation, I study runtime safety filters for modular systems that use uncertainty quantification. I argue that uncertainty quantification, while ill-defined for standalone modules, should be contextualized with respect to how the uncertainty measures are *used* by the rest of the system. I present novel analysis techniques with which one can compare uncertainty-aware system designs, produce specifications on uncertainty measures, and examine system vulnerabilities, and apply the techniques to real-world systems.

In the second part of this dissertation, I examine how to apply statistical verification techniques for control problems. I present three methodologies. In the first, I show how conformal prediction can be used in a safe planning framework in a dynamic environment. In the second, I present a novel technique to perform active learning to calibrate a learned reachable set while simultaneously producing a probabilistic generalization bound. Finally, I introduce a scenario optimization-based hierarchical framework used to guide a deployed controller towards regions that are most likely to be safe. All three methodologies create a bridge between statistics and control, balancing assumptions needed by different tools and bringing to light potentially unintuitive design tradeoffs.

Contents

Contents **i**

List of Figures **iii**

List of Tables **viii**

1 Introduction **1**
1.1 Runtime Safety Filtering 2
1.2 Statistical Verification 5

I Runtime Safety Filtering **9**

2 Case Study System Descriptions **10**
2.1 Autonomous Driving System 10
2.2 Runway Incursion Detection System 12

3 Measuring System Robustness **15**
3.1 Introduction 15
3.2 Problem Statement 15
3.3 Method 17
3.4 Results on Case Studies 18

4 Generating Specifications on Module Uncertainty **23**
4.1 Introduction 23
4.2 Related Work 25
4.3 Definitions and Assumptions 26
4.4 Problem Statement 27
4.5 Generating Specifications on Uncertainty Measures 28
4.6 Conclusion to Part I 39

II Statistical Verification **40**

5 Multi-Agent Reachability Calibration with Conformal Prediction **41**
5.1 Introduction . . . 41
5.2 Related Works . . . 43
5.3 Assurances from Uncertainty . . . 44
5.4 Probabilistic Reachability and Planning . . . 47
5.5 Results . . . 49
5.6 Discussion and Future Work . . . 53
5.7 Appendix . . . 54

6 Active Calibration of Reachable Sets Using Approximate Pick-to-Learn **59**
6.1 Introduction . . . 59
6.2 Related Work . . . 61
6.3 Problem Definition . . . 61
6.4 Approximate Pick-to-Learn . . . 62
6.5 Simulation Experiments . . . 65
6.6 Conclusions . . . 68
6.7 Appendix . . . 69

7 Hierarchical Probabilistic Verification for Reachability Learning **75**
7.1 Introduction . . . 75
7.2 Preliminaries . . . 77
7.3 Hybrid Reachability Verification Method . . . 78
7.4 Theoretical Reach-avoid Guarantees . . . 82
7.5 Experiments and Results . . . 84
7.6 Conclusions . . . 86

III Conclusion **87**

8 Discussion and Future Directions **88**
8.1 Advances and Limitations . . . 89
8.2 Directions for Future Work . . . 89

Bibliography **92**

List of Figures

2.1 The sequential, modularized system architecture, in which x is the state, u is the control, and $C(u)$ represents a cost function evaluating the performance of the overall pipeline. Module i has output y_i with associated uncertainty σ_i. The terms in italics, "Perception", "Planning", and "Control", represent example functionalities of the modules. 11
2.2 The Autonomous Driving system, with an example scene from the nuScenes dataset. The ego agent must navigate to its goal in the presence of other agents, of which only the nearest agent is shown. 12
2.3 The Runway Incursion Detection system, with an example image that might be input to the detector on the aircraft. From this aerial view of the runway, the aircraft must determine if there exist ground vehicles which could cause a runway incursion. Red circles correspond to detections. Yellow circles correspond to "probationary" tracks that are not fully initiated yet. Blue circles correspond to fully initiated tracks. The numbers correspond to detection confidence scores at that instant in time. 13

3.1 We propose to measure a system's input-output robustness by searching for all input errors such that the system output incurs an evaluation cost below a threshold. In the example results shown, we display functions and level sets that are the solution to the search problem. Different colors and line styles indicate different system designs. The light green color indicates evaluation on the Runway Incursion Detection system and the dark green color indicates evaluation on the Autonomous Driving system. 16
3.2 (Runway Incursion Detection System Case Study) We plot the fitted Gaussian Processes representing $F(\epsilon)$ for three different cost functions: the miss rate (left), false positive rate (middle), and MOTP (right). Thus, here "noise level" refers to the ϵ brightness perturbation value. Computing the sub-level sets of these functions yields a one-dimensional interval, which we then compute the size of by counting grid points. 18

3.3 (Autonomous Driving System Case Study) We show the sub-level sets of $F(\epsilon)$ for the holistic (left) and safety (right) cost functions. For the holistic cost, $c = 0.05$, and for the safety cost, $c = 0.02$. We omit sub-level sets of size 0. Some sub-level sets occupy the whole space. Part of the sub-level set for the Baseline system for the safety cost is also shown in Figure 3.1. 21

4.1 We propose that generating a specification on a module's uncertainty measure will help us to apply our system-level perspective. Using assume-guarantee contract theory, we treat this as a problem of finding the contract of an upstream module of interest (Contract 1) given the system contract and the contract of the downstream decision-making modules (Contract 2). We show some example results in which we use data to generate histograms that help to characterize the Contract 2. The light green color indicates evaluation on the Runway Incursion Detection system and the dark green color indicates evaluation on the Autonomous Driving system. 24

4.2 (Running Example) (Top) A plot showing the raw data used to estimate $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ for when the prediction error is below the threshold and the ego agent and target agent are close together. (Bottom) Plots showing one of the scenarios in the dataset with the associated plan for three different settings of the uncertainty value. Since this is the agent-avoiding planner, we indeed see that as the uncertainty value increases, the plan (dark blue solid line) takes a more conservative maneuver to reach the goal. The evaluation cost here is the safety cost. 31

4.3 (Running Example) For the agent-avoiding planner, we show the final $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values for (left) evaluation using the safety cost and (right) evaluation using the holistic cost. For each plot, we set the values of $\mathbf{1}(e_1 < e_{thres})$ and z and show how the probability values vary by the uncertainty value. The value of e_{thres} is 1.5, the value of c is 0.02, and a low ego-agent distance is defined as a distance less than 5 meters, while a high ego-agent distance is a distance greater than 5 meters. In general, these values can be chosen arbitrarily by the system designer. As the uncertainty value increases, the agent-avoiding planner keeps the ego agent further away from other agents, which can make the ego safer but worse at holistic driving. 33

4.4 (Autonomous Driving System Case Study) For the lane-keeping planner, we show the final $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values for (left) evaluation using the safety cost and (right) evaluation using the holistic cost. For each plot, we set the values of $\mathbf{1}(e_1 < e_{thres})$ and z and show how the probability values vary by the uncertainty value. Again, the value of e_{thres} is 1.5, the value of c is 0.02, and a low ego-agent distance is defined as a distance less than 5 meters, while a high ego-agent distance is a distance greater than 5 meters. As the uncertainty value increases, the lane-keeping planner's cost does not monotonically increase for either the safety or holistic costs, since the planner is using the uncertainty value to trade off between different objectives. 35
4.5 (Runway Incursion Detection System Case Study) We show the data used to estimate the $P(C(u) > c|l_1)$ values for the Naive Tracker (a), Regression Tracker (b), and Histogram Tracker (c). Each point represents a track, which corresponds to a sequence of detector confidence values. These points are binned along the Calibration Error (l_1) axis. (d) Plot showing the $P(C(u) > c|l_1)$ values for the Naive, Regression, and Histogram trackers side-by-side. The lowest bars represent a value of 0. The value of c is 0.4 and the detector calibration errors have bin sizes of 0.1. 38

5.1 Our method generates dynamically-feasible, probabilistic confidence sets that are derived from a conformal-calibrated quantile regression model. 42
5.2 Visualization of the running example. The autonomous ego vehicle is shown in red, and the human driver is shown in blue. The ego vehicle aims to navigate to the pink star while avoiding a collision with the human-driven vehicle. Confidence sets for the next three prediction steps are shown. In (b), the redder regions represent confidence sets for earlier prediction timesteps, and the translucent regions represent conformal prediction's calibration effect. 45
5.3 Case Study of Uncertainty Metrics. We demonstrate a simple example in which the choice of uncertainty measure affects the size of sets, with coverage rate held constant. 53
5.4 Our algorithm is applied to assure safety in potential runway incursion scenarios. Once the ground vehicle is determined to have crossed a designated safety threshold, the aircraft is cleared to land. 54
5.5 Visualization of the planner. The autonomous ego vehicle is shown in red, and the human drivers are shown in blue. The plan is shown in green, representing the tracking of the original HJ-generated plan using iLQR. 57

6.1 (a) Our framework provides a probabilistic generalization bound for the *iterative* process of learning and actively sampling the next data point. The novelty of our method lies in the realization that if we conformally calibrate our active sampling strategy so that it approximates maximizing the true model error, we can achieve a desired probabilistic guarantee. (b) We demonstrate our method on a drone racing example in which the ego drone (green) overtakes another drone (yellow). We display the reach-avoid set learned by our method, with the lighter colors indicative of the set at previous iterations and the darker colors for later iterations. The orange points are samples taken by our calibrated active learning strategy. 60

6.2 (a) In this 2-dimensional "Slice 1", the ego drone's 3D velocity is set to $[0.0, 0.7, 0.0]$ and its altitude is 0.0. The other drone's 3D spatial coordinates are $[0.4, -2.2, 0.0]$ and its 3D velocity is $[0.0, 0.3, 0.0]$. (b) In this 2-dimensional "Slice 2", the other drone's state is the same, but the ego drone's velocity is $[0.0, 0.0, -0.5]$ and its altitude is 0.05. The baselines' calibration technique is to simply choose an appropriate level of the learned value function. For the more commonly seen scenario (a), the level sets have more regular shapes, better justifying this calibration technique. However, for the less commonly seen scenario (b), this is not the case, and it is more reasonable to be unconstrained by learned set geometries, as in our method. In both figures, orange points are the samples that our method took. 66

6.3 Plots comparing the number of samples and FPR/FNR across methods. "Boundary" denotes our boundary sampling active learning technique, while "Random" refers to setting $a_h(x)$ to random values. In 3D experiments, the slice of the dynamics is the same as that for the 2D Slice 1 except the ego drone's altitude is also allowed to vary. In 4D experiments, we additionally allow the ego agent's velocity in the x-direction to vary. The numbers above the bars indicate how many seeds out of 10 were successful for our method. Our method balances the competing objectives of minimizing the number of samples and FPR and FNR well compared with the baselines. As we expect, performance for all methods is worse for Slice 2 compared with Slice 1. 68

7.1 (a) Building on a learned reachability value function and policy from prior work (e.g., [99]), our hierarchical certification-guided framework uses probabilistically certified global and local safe sets to guide decision-making, invoking *safe set recovery* to drive the agent to a certified region. Our key novelty is leveraging scenario optimization for local safe set expansion. (b) Drone racing example in which the ego drone (blue) reaches a refined locally certified safe set which empirically results in safe overtaking of an opponent (black). 76

7.2 Evolution of a drone racing simulation against an opponent (black), comparing our method (Hybrid, dark blue) with various baselines. Baselines struggle to balance safety and goal-reaching, whereas our method achieves both, safely overtaking and reaching the gate. Moreover, we observe that the ego agent leverages safe recovery MPPI to try to enter the locally refined set. 85
7.3 (a) Performance of our framework compared to control barrier function (CBF) and soft-constraint baselines. (b) Ablation study comparing the hybrid method to its learning-only and model-based variants. In both (a) and (b), the results for our method are shown for reference. *Our framework achieves higher success probability over 500 sampled initial conditions, effectively balancing collision avoidance and goal-reaching.* 86

List of Tables

3.1 Sub-Level Set Sizes and Costs for Tracker Designs 19
3.2 Sub-Level Set Sizes and Costs for AV Designs 22

5.1 Coverage Rates and Set Sizes for $1 - \gamma = 0.95$. 50
5.2 Waymo Planning Benchmarks 52
5.3 Un-Calibrated Confidence Set Coverage Metrics 57
5.4 Un-Calibrated Confidence Set Sizes 57

6.1 Comparing $\bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha$ and ϵ_{LB} values to understand the tightness of the probabilistic bounds - lower values generate stronger statements. "LB Robust, Min ϵ" always has the lowest ϵ, but it uses the most samples. 69

Acknowledgments

I owe an enormous debt of gratitude to many folks for not only helping me make it to this point, but also helping to shape me into the person I am. I feel incredibly fortunate and honored that they were the ones who I met and who were with me in this journey.

My first thanks goes to my advisor, Professor Claire Tomlin. Professor Tomlin has known me ever since the beginning of my undergraduate years and has been a very formative influence in my life, creating the right conditions for and guiding my growth. If I have any understanding of what good science or a good scientist should look like, it is because of her. I hope one day to have her level of wisdom, compassion, patience, and foresight, and I am deeply honored to have been her student.

I also want to thank the rest of my committee. Professor Shankar Sastry has been one of my big supporters. His advice, both in research and career development, is profound and thought-provoking, and his high-level vision and guidance has served to give my research the right direction. He has pushed me towards greater excellence and mathematical rigor while simultaneously being very generous and understanding. Professor Sanjit Seshia taught me about formal methods and provided important perspectives, pointers, and corrections in my work. He has been very kind and helpful in all of his support. Professor Marco Pavone broadened my thinking with his invaluable perspectives and ways of foreseeing the next important problems. He has also been a wonderful source of feedback and advice.

I would certainly not have begun a PhD, much less have made it to the end, without the support of many other mentors that I had throughout my undergrad and grad school. The now-Professors Sylvia Herbert, Andrea Bajcsy, David Fridovich-Keil, Somil Bansal, Andreea Bobu, and Jaime Fernandez-Fisac were my grad student mentors when I was an undergrad, and have continued to give me advice during my PhD. I feel extremely fortunate that they took me under their wing when I was getting started in research, patiently teaching me various fundamentals, being my role models, believing in my potential, and giving me opportunities. I also want to thank Professor Gireeja Ranade for teaching me the value of teaching and for telling me that "great ideas are simple". I am grateful to Dr. Iman Soltani, under whose guidance I worked on my first independent research project, and also Professor Anca Dragan for her inspiring teaching. Finally, I would like to thank Dr. Rohan Chitnis for some crucial pieces of advice that helped me to find my path.

I had the pleasure to work with many very talented collaborators during my PhD: Michael H. Lim, Johnathan Tucker, Zachary Sunberg, Aleksandra Faust, Anish Muthali, Haotian (David) Shen, Rebecca Roelofs, Dragos Margineantu, Boris Ivanovic, Peter Karkus, Ebonye Smith, Jingqi Li, and Gechen Qu. Some of these folks were my mentors as well. I especially want to thank Michael Lim, who mentored me during the first few years of my PhD and from whom I learned so much about how to do research. I am very grateful to not only have learned a great deal from all of my collaborators, but also, and more importantly, have enjoyed working with them.

I would like to thank Chih-Yuan Chiu as well for helping me with prelims and with teaching linear systems. I am also grateful to all of the folks at Nuro who I learned from

during my internship there. Additionally, I had the opportunity to organize several seminars and workshops during my PhD, which were both fun and intellectually enriching, and I would like to thank all of my co-organizers: Chinmay Maheshwari, Pan-Yang Su, Bike Zhang, Nima Rahmanian, Ebonye Smith, Hanna Krasowski, all of the Out-of-Distribution Generalization in Robotics workshop team, all of the Safety and Security in GenAI workshop team, and many other wonderful folks in the SCALE Lab at Berkeley.

I am extremely lucky to have had amazing labmates in the Hybrid Systems Lab: Chris Strong, Ian Chuang, Sara Pohland, Jingqi Li, Jason Choi, Anand Siththaranjan, Chams Mballo, Ebonye Smith, Kaylene Stocking, Michael Lim, Gabriel Colón Reyes, Marsalis Gibson, Anish Muthali, David Shen, and all other past and current members. My labmates have greatly inspired me to learn and push myself, given me invaluable advice and feedback, sparked wonderful discussions, and have been great friends in addition. I especially want to thank Anand, Chris, and Kaylene (we were desk neighbors) for all of the fun hangouts and discussions. A big thanks also goes to Jessica Gamble, Shirley Salanio, Judy Smithson, and all of the administrative staff who work so hard to support all of us grad students. Thanks to other great friends for all of the food excursions and good memories: Sukanya Kudva, Victoria Tuck, Ricardo Sandoval, Neelay Junnarkar, Arwa Alanqary, Zhe Fu, Simeon Adebola, and Hanna Krasowski. A special shout-out goes as well to all of the WiCSE folks for creating such a nice community.

Finally, I want to thank my family. To my parents, Aniruddha and Deepali Deglurkar, for their selfless love and unwavering belief in me. They always put me first no matter what. To my grandparents, Ramesh and Ranjana Ghate, for all of their blessings, and especially to my grandfather, who I greatly miss and who I know would have loved to see me complete this milestone. Finally, to my sister Shivani for being my best friend and confidante. This dissertation belongs to you all.

Chapter 1

Introduction

Modern autonomous systems, ranging from robots to autonomous vehicles to aviation systems, are often machine learning-based, which enables them to operate in more general and unstructured environments but in turn also increases their complexity. This means that the systems are susceptible to new kinds of dangerous failures. For classical, non learning-enabled systems, previous literature introduced the field of Fault Detection, Isolation, and Recovery (FDIR) [19, 80] to prevent potential failures during system operation. FDIR provides tools for *detecting* failures before they occur during system operation. *Isolation* helps create an understanding of which portion of the system may be contributing to the failure, and *recovery* techniques can then ensure that the system remains performant even in the presence of the fault. On the other hand, the fields of system *verification* and *validation* produce proofs that a system will avoid failure and methodologies to test a system in its operational domain and analyze it to ensure it is safe [52, 91].

In the classical context, FDIR often consists of generating residuals that capture the difference between expected and true system behavior, and tracking the residuals over time, for example using an observer or Kalman filter. Multiple such residuals can be maintained over time to create a fault signature and identify which system component faulted. Finally, recovery is often performed through hardware redundancy and fault-tolerant control. Meanwhile, system verification draws from techniques from the formal methods literature such as model checking [51] and reachability analysis [7]. Model checking techniques determine if a system model satisfies a specification and provide counterexamples if not, and reachability analysis can be used to determine system states from which the system execution is guaranteed to be safe or unsafe. Finally, system validation and testing often involve simulating or deploying the system in many different conditions to find failure cases.

Many of these principles remain relevant for learning-enabled systems. However, learning-enabled systems present new challenges for FDIR and verification. These systems typically do not have interpretable and trustworthy models that can be used for traditional estimation and control. Since they are designed for complex environments, it can be difficult to define desired behavior in the first place. Additionally, many machine learning models are extremely large, with millions or even billions of parameters, making model checking

or reachability analysis computationally intractable [137]. Machine learning is also often used for *semantic* understanding and high-level reasoning, such as for performing perception or language-related tasks [95]. In such cases, system failures also take on more semantic meaning, which makes them more difficult to detect and diagnose.

This dissertation considers both FDIR, which I henceforth call *runtime safety filtering*, and verification/validation for learning-enabled autonomous systems. Both of these together I call *assurance frameworks*. Part I of this dissertation concerns runtime safety filtering, and Part II concerns *statistical verification*; that is, the production of probabilistic safety certificates by testing the system in different scenarios. In the formal methods literature, runtime safety filtering is often termed *runtime assurance* [42, 150, 164], and statistical verification is done by works performing *statistical model checking* [3, 50, 163]. In the next two sections, I explain the runtime safety filtering and statistical verification problem domains, detailing the challenges motivating this dissertation. In each section, I give an overview of the key contributions.

1.1 Runtime Safety Filtering

Since part of the runtime safety filtering paradigm involves detecting and isolating faults in the system before a failure occurs, in this section I focus on *modular* software pipelines for autonomous systems. In the modular architecture (as opposed to the end-to-end framework), different functionalities are processed, usually in a sequence going from sensing to control, by different components. For example, a robot may be tasked with navigating to a target location in an environment using a camera as a sensor. One classic software architecture for this system would consist of a perception module that processes images into a representation, such as bounding boxes and labels for objects in the environment. This output can then be used by a planner module to plan a trajectory for the robot to reach its goal while avoiding collisions with the objects in the environment. Finally, a control module would use this plan to produce actuator commands that enable the robot to follow the plan [14, 30].

One advantage of the modular architecture is its amenability to fault detection and recovery; if one can check the output of each individual module and ensure that it is correct, system failures can be averted. However, when these modules are machine learning-enabled, this becomes very challenging. Instead, a common paradigm is to rely on *uncertainty quantification* [62]. That is, each machine learning-enabled module can produce an *uncertainty measure* indicating "how confident" the module is in its output. This serves as a proxy for fault detection; if a module is highly uncertain, then it should be the case that it is likely the module is wrong. The subsequent modules can then incorporate this uncertainty measure into their decision-making [47, 71, 83, 111]. This serves as a proxy for fault recovery; the downstream modules can react to the upstream uncertainty measure in such a way so as to make sure the system remains safe overall. The term *runtime safety filtering* thus comes from the attempt to "filter out" unsafe system outputs deriving from a fault.

Both uncertainty quantification and understanding how to use an uncertainty measure

to ensure safety are hard problems. Moreover, they are *coupled* problems within the overall design of the system. Part of the reason for this is that uncertainty can be a very ill-defined concept, as I explain in the next section. Additionally, decision-making under uncertainty in its full generality is computationally intractable, even if the uncertainty is well-defined [127]. This has led many practical, real-world systems to rely on heuristics or otherwise neglect uncertainty quantification altogether.

Uncertainty Quantification

Many philosophers have studied knowledge and uncertainty since at least the time of Socrates, from whom we derive the concept of Socratic ignorance, or knowing the limits of one's own knowledge [128]. Philosophers studying metacognition and higher-order knowledge, or what we can know about our own knowledge, disagree on whether one always knows that one knows something, or even whether one can always find a justification for one's own knowledge [20, 36]. In fact, the contemporary philosopher Timothy Williamson has argued that humans do not always know their own mental states, whether they are knowledgeable, ignorant, certain, or uncertain (an argument for so-called *anti-luminosity*) [160]. Given how much we model machine intelligence after our own, it makes sense that designing a module that "knows" when it could be wrong would be similarly difficult.

When we discuss *quantifying* uncertainty, we are in fact relying on a Bayesian perspective: one may have a *degree* of belief that any given proposition is true. Such a degree of belief can be interpreted as a confidence in the belief and it may change given evidence. These Bayesian principles have been applied for models such as Bayesian neural networks [123] and Gaussian Processes [131], which maintain distributions over their respective predictive functions f given the *evidence* of the data D. Given a way to calculate the probability distribution $P(D|f)$, one can use Bayes' Rule to update $P(f|D)$. The model can then use $P(f|D)$ to produce a *prediction distribution* instead of only a point prediction, for example by computing $P(y|x, D) = \int P(y|x, f)P(f|D)df$ for inputs x and outputs y. $P(f|D)$ thus can be an expression of how certain the model is that f is the "correct" predictive function given the data.

Unfortunately, for Bayesian neural networks the computations of $P(f|D)$ and $P(y|x, D)$ can be intractable, especially for large neural networks. This has motivated a slew of uncertainty quantification techniques aiming to approximate the computations, for example using Monte-Carlo dropout [63], model ensembling [94], and variance propagation [130]. However, even if it were possible to obtain these computations exactly, additional points of consideration remain: how should one select the prior distribution over f? What form should the distribution $P(D|f)$ take? What is a tractable uncertainty measure, given the prediction distribution, that can be used by downstream modules? Many of these questions are often left up to modeling assumption or design choice [117].

Additional complications arise from distinctions made in the literature between *epistemic* and *aleatoric* uncertainty [89]. While epistemic uncertainty is stated to be "model" uncertainty, or uncertainty that disappears given infinite training data, aleatoric uncertainty is

"inherent", irreducible uncertainty present in the problem domain or environment. Disentangling these two uncertainties is also a subject of concern [31]. More recently, terms such as "out-of-distribution" or "distribution shift" have become more dominant in the community to describe errors in machine learning models [140]. These terms are meant to refer to data points "outside" the training data distribution of the model, although what "outside" means is not defined in a principled manner, and a model can theoretically perform badly even "within" the training data distribution [55, 69, 100, 161]. Various techniques exist for out-of-distribution detection, including examining energy scores [108], computing distances in the feature space [144], gradient-based techniques [139], using softmax probabilities [101], and so on. A similar theme arises here in that, given the difficulty of defining and computing uncertainty, a practical system will simply select some uncertainty quantification technique for a particular task based on observed performance and general engineering constraints [2, 135].

Core Contributions

Due to the difficulty of performing uncertainty quantification and decision-making under uncertainty, the two are often left as design choices. Given this, in Part I of this dissertation I ask what makes a good uncertainty-aware system design and present novel analysis techniques for existing designs. The goal is to produce guiding principles for engineering decisions.

- (Chapter 2) Case Study System Descriptions: In order to produce grounded analysis techniques and motivate my work further, I first present two real-world and complex autonomous systems that serve as case studies. The first system is an autonomous driving system consisting of a learning-enabled predictor and classical planner and controller. The second system is an aircraft runway incursion detection system consisting of a computer vision-based detector and an object tracker.

- (Chapter 3) Measuring System Robustness: The first analysis technique presents a robustness metric for an autonomous system, measuring how the system performs under different amounts of distribution shift. Essentially, we sample different perturbations to the input of the system and take the expected system cost for each perturbation. A more robust system is then defined as having a low expected cost for more perturbation amounts. The robustness metric is used to compare uncertainty-aware and baseline system designs and study how performance and robustness trade off. This work attempts to make "distribution shift" a more concrete term that can help a designer to understand system vulnerabilities.

- (Chapter 4) Generating Specifications on Module Uncertainty: The second analysis technique probes at the heart of uncertainty quantification and how it can be useful if it is so ill-defined. The main argument here is that an uncertainty measure is given meaning by how it is *used* by the downstream modules in the system. With this

insight, we present a framework to generate a specification on a module's uncertainty measure, given a system-level specification. Doing this requires one to characterize how well the downstream modules use the uncertainty measure. Thus, we simultaneously inform a designer on how much effort is needed to produce an uncertainty measure that accurately predicts the module error and provide a way to compare decision-making-under-uncertainty algorithm designs.

1.2 Statistical Verification

Statistical verification, as I call it in this dissertation, relates to blending classical notions of verification and validation to produce assurances for machine learning-enabled systems. The essential idea is that, by repeatedly testing a system in different scenarios and thus obtaining *samples* of system behavior, one can obtain a probabilistic assurance on the system error or cost in new situations. This may be a mathematical statement similar to $P(\textit{system error} < \textit{threshold}) > 1 - \epsilon$ for a small value ϵ.

While in the previous section on runtime safety filtering I discuss problems that arise when the autonomous system is *modular*, such as the difficulty of performing uncertainty quantification, statistical verification techniques can be performed for either modular or end-to-end systems. Indeed, in recent years more autonomous systems have become end-to-end and even modular systems have become more complex [32, 75, 158]. The inherent difficulty of performing uncertainty quantification and decision-making under uncertainty has only been exacerbated by the advent of foundation models and multi-modal models. Even for simpler systems, uncertainty measures often have need of *calibration*, which relies on the statistical techniques discussed here (see Chapter 5 of this dissertation). Statistical verification has therefore become a very important and attractive research direction.

The design of probabilistic certificates involves various parts that are intertwined in trade-offs. These include the number of samples required, the strength of the guarantee (i.e. the tightness of the probabilistic bounds), the assumptions on the distribution the samples are drawn from, and the system error definition. Assumptions on the sample distribution perhaps play the greatest role in this dissertation, as I question how to use existing statistical tools while relaxing those assumptions. Indeed, this raises deeper questions on how we can produce probabilistic assurances in the first place – where did the samples used to produce the assurance come from, and how will situations the system encounters in the future compare? After all, a system may encounter with any catastrophic, highly unlikely, and unpredictable failure during deployment that assurance frameworks were unable to capture (a *black swan event*, according to statistician Nassim Nicholas Taleb [146]). While the field of statistical verification does ultimately need to contend with these questions, in this dissertation I focus first on applying existing tools to autonomous control problems and understanding the interface between statistics and control when machine learning is involved.

Overview of Key Techniques

In this section, I give a brief overview of two techniques used to produce the type of Probably Approximately Correct (PAC)-style guarantees [138] that will be present in this dissertation. These are (1) conformal prediction and (2) scenario optimization.

Conformal Prediction. Conformal prediction is a distribution-free statistical tool that has gained popularity in the learning and control community as a way to calibrate the outputs of a model while providing probabilistic assurances [9, 106, 154]. In this dissertation, I primarily focus on what is known as *split* conformal prediction. In this paradigm, one maintains a *calibration* dataset that is separate from the training and testing datasets for the model. A *score function* is also required that evaluates the model outputs that need to be calibrated. For example, for a model f with input x, one possible score function $s(x, y)$ can be $s(x, y) := \|y - f(x)\|$; that is, the distance between $f(x)$ and the true prediction y. Let $R^{(i)}$ be defined as the evaluation of the score function for the i^{th} data point in the calibration set, with N points total in the calibration set. Let R^{test} be the evaluation of the score function for a test data point. Then, what is known as a *marginal* conformal prediction guarantee states that

$$P_{R^{test},R^{(1)},R^{(2)},...R^{(N)}}\left(R^{test} < Quantile_{\lceil (N+1)(1-\delta)\rceil/N}(R^{(1)}, R^{(2)}, ...R^{(N)}, \infty)\right) \geq 1 - \delta$$

when it is assumed that $R^{test}, R^{(1)}, R^{(2)}, ...R^{(N)}$ are *exchangeable* random variables (that is, their joint distribution remains the same no matter how they are permuted). This statement serves as a generalization bound in the sense that, if R^{test} is exchangeable with the points in the calibration set, meaning that the calibration set is reflective of the points that will be seen at test time, the probability that it is above a threshold is upper bounded.

This overall procedure can be used for calibration by designing the score function appropriately. One common example is to design an interval around the nominal model output and define the score function as a distance of the true prediction from that interval. Then, one can compute the quantile of that score function on the calibration dataset and set the size of the predicted intervals such that a test point lies within the interval with high probability.

A second type of guarantee is obtained via *conditional* conformal prediction, in which conditioning occurs on $R^{(1)}, ...R^{(N)}$:

$$P_{R^{test}}\left(R^{test} < Quantile_{\lceil (N+1)(1-\delta)\rceil/N}(R^{(1)}, R^{(2)}, ...R^{(N)}, \infty) \mid R^{(1)}, R^{(2)}, ...R^{(N)}\right) \sim Beta(N + 1 - l, l)$$

for $l = \lfloor (N + 1)\delta \rfloor$ and with the same exchangeability assumption. In other words, the probability that R^{test} is upper bounded by the quantile follows a Beta distribution because the R values in the calibration set are random variables. So, conditional conformal prediction can yield a two-layer probabilistic statement:

$$P_{R^{(1)},R^{(2)},...R^{(N)}}\Bigg[P_{R^{test}}\left(R^{test} < Quantile_{\lceil (N+1)(1-\delta)\rceil/N}(R^{(1)},R^{(2)},...R^{(N)},\infty) \mid R^{(1)},R^{(2)},...R^{(N)}\right) \geq 1-\delta\Bigg] \geq 1-\beta$$

where β is calculated by taking the appropriate mass of the Beta distribution that is less than $1-\delta$. This is a PAC guarantee that is useful for calibration, statistical verification, and the safe design of autonomous systems.

Scenario Optimization. Scenario optimization refers to solving chance-constrained optimization problems using samples and then providing probabilistic guarantees that the obtained optimal solution indeed satisfies the constraints [28]. Specifically, for an optimization problem of the form

$$\begin{aligned} &\min_{z\in\mathcal{Z}} c^\top z \\ &s.t.\ P_\Delta(h(z,\Delta)>0) \leq \epsilon \end{aligned}$$

where $\mathcal{Z}$ is a convex and closed set, h is a convex function, and Δ is a random variable, scenario optimization takes samples of Δ to convert the probabilistic constraint into N deterministic constraints, where N is the number of samples:

$$\begin{aligned} &\min_{z\in\mathcal{Z}} c^\top z \\ &s.t. \bigwedge_{i=1}^{N} h(z,\delta^{(i)}) \leq 0 \end{aligned}$$

Here, the $\delta^{(i)}$'s are the independent and identically distributed (IID) samples. Let z_N^* be the solution to the scenario program. Then, if $N \geq \frac{2}{\epsilon}\left(\ln\frac{1}{\beta}+d\right)$, we obtain the PAC guarantee

$$P_{\delta^{(1)},...,\delta^{(N)}}\Big[P_\Delta\left(h(z_N^*,\Delta)>0\right) \leq \epsilon\Big] \geq 1-\beta$$

for d the dimension of the decision variable z and β a confidence parameter pertaining to the outer probability over the samples taken. This is also a "generalization bound" in the sense that, if one is given a new sample δ, which is IID with $\delta^{(1)},...,\delta^{(N)}$, and evaluates $h(z_N^*,\delta)$, the probability that the constraint is violated is upper bounded.

This probabilistic guarantee appears very similar to the probabilistic guarantee given by conformal prediction, especially conditional conformal prediction. This is not a coincidence,

and there is a strong relationship between scenario optimization and conformal prediction, as several recent works have shown [45, 104, 125]. In this way, scenario optimization also proves to be a powerful tool for statistical verification. It is additionally useful compared to conformal prediction in that it can *produce* a minimizer of some objective *as well as* probabilistically certify it.

Core Contributions

- (Chapter 5) Multi-Agent Reachability Calibration with Conformal Prediction: This work transitions between Part I (Runtime Safety Filtering) and Part II (Statistical Verification) by applying tools such as conformal prediction for producing an assured autonomous system pipeline. In this work, we connect uncertainty quantification performed for a neural network that predicts the motion of other agents with the downstream planning and control performed by an ego agent so that the overall system is safe. Conformal prediction is used to calibrate the predictor's uncertainty measure and produce a probabilistic guarantee. In this setting, the main concern is that the samples used for conformal prediction are not exchangeable. We circumvent this assumption by using the Rolling Risk Control framework [56], which allows for safety in dynamic and online settings.

- (Chapter 6) Active Calibration of Reachable Sets Using Approximate Pick-to-Learn: We again examine the assumptions and trade-offs involved in statistical verification by studying verification in dynamic situations. We consider the setting of a learned model of a reachable set. Since this model may have errors, we apply an active learning technique to acquire samples where the model is likely to be wrong and then re-fit the model. Since the samples are arising from a learning process, they cannot directly be used for statistical verification without further modification. In this work, we apply the previous algorithm Pick-to-Learn [126] to this scenario, modifying it to suit the active learning domain, and merge it with conformal prediction to produce a generalization bound.

- (Chapter 7) Hierarchical Probabilistic Verification for Reachability Learning: Finally, in this work we apply scenario optimization to perform statistical verification of a learned policy in a reachability setting. The key idea here is to combine two different probabilistic certificates in a hierarchical way. That is, while one certificate may only provide guarantees for one part of the state space, the other certificate can build off of the first to certify additional parts of the state space.

The contents of this dissertation are based on work with Haotian Shen, Anish Muthali, Marco Pavone, Dragos Margineantu, Peter Karkus, Boris Ivanovic, Michael H. Lim, Rebecca Roelofs, Aleksandra Faust, Ebonye Emith, Jingqi Li, Gechen Qu, and Claire J. Tomlin.

Part I

Runtime Safety Filtering

Chapter 2

Case Study System Descriptions

In this chapter, I give an overview of two real-world, complex, and industry-grade autonomous system pipelines that are analyzed in Part I of this dissertation. These are both *modular* systems for which uncertainty quantification will play a large role in the runtime safety filtering. The notation and terminology introduced in this chapter will be used in Chapters 3 and 4.

Both of these systems take on a general structure that is represented in Figure 2.1. The system as a whole maps a state input $x \in \mathcal{X}$ to a control command $u \in \mathcal{U}$. Each module is represented as a function f_i that maps the previous module's output $\{y_{i-1}, \sigma_{i-1}\}$ to its own output, $\{y_i, \sigma_i\}$. Here, y_i is the module's nominal output and σ_i is the uncertainty measure associated with that nominal output. These definitions are given more mathematical formality in Chapter 4. Here, I only hope to make this architecture more concrete to the reader via the examples of the case studies.

This chapter is based on work with Haotian Shen, Anish Muthali, Marco Pavone, Dragos Margineantu, Peter Karkus, Boris Ivanovic, and Claire J. Tomlin in [40] and [41].

2.1 Autonomous Driving System

One system that we analyze in Part I is the Diffstack autonomous driving software pipeline, introduced by NVIDIA Research in [87]. We use the nuScenes dataset [24], which contains real driving data from Boston and Singapore. This system is represented in Figure 2.2.

This is a 3-module system comprised of a trajectory forecaster for other agents in the environment (f_1), a sampling-based planner (f_2), and a controller using MPC (f_3). Here, the trajectory forecaster is the Trajectron++ model [132], a recurrent model designed for autonomous driving which takes, among other inputs, non-ego agent state histories and outputs a Gaussian Mixture Model (GMM) representing the distribution of a non-ego agent's future control actions, which are converted into trajectories given the agent's dynamics. This distribution can help to capture uncertainty heuristics about the non-ego agent's future motion. For example, we may take the uncertainty measure of the predictor to be (i) the

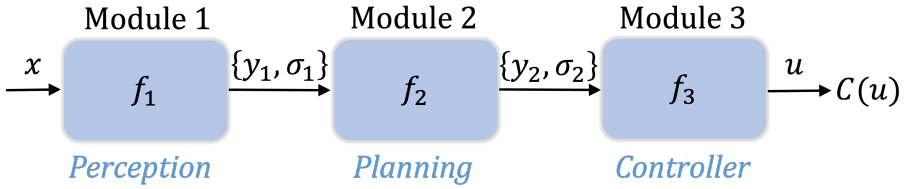


Figure 2.1: The sequential, modularized system architecture, in which x is the state, u is the control, and $C(u)$ represents a cost function evaluating the performance of the overall pipeline. Module i has output y_i with associated uncertainty σ_i. The terms in italics, "Perception", "Planning", and "Control", represent example functionalities of the modules.

entropy of the most-likely Gaussian of the GMM, which we denote as σ_{ML}, or (ii) the entropy of the categorical distribution over the GMM's modes, which we write as σ_K. These values are normalized to lie in $[0, 1]$.

The planner takes candidate trajectory samples and outputs the trajectory sample that minimizes an internal cost function C^{int}, as seen in [87]:

$$C^{\text{int}}(\cdot; w) = w_1 C_{\text{coll}}(s, \hat{s}) + w_2 C_{\text{goal}}(s, g) + w_3 C_{l\perp}(s, m) + w_4 C_{l\measuredangle}(s, m) + w_5 C_u(s) \tag{2.1}$$

where s represents the trajectory sample, $\hat{s}$ represents the trajectory modes of the GMM prediction distribution along with their probabilities, g is the goal, and m is the lane graph. Each term denotes the cost of collision, the distance to the goal, the lane lateral deviation, the lane heading deviation, and the control effort, respectively. The w values are weights. Finally, the MPC controller outputs the final ego agent trajectory by solving an iterative LQR problem optimizing this internal cost, using the planned trajectory as a starting point for the optimization. Following [87], we evaluate the predictions only for the non-ego agent that is closest to the ego agent at the current time, which we will call the *target* agent.

Thus, we define x to be the history of the target agent for prediction as well as the overall environmental scenario, for example relating to agent relative configurations or road types. The lengths of the target agent's history and prediction horizon are the same as those in the Trajectron++ work. We have that y_1 is the $\hat{s}$ in Equation (2.1) and σ_1 is the uncertainty measure that is extracted from the GMM. The trajectory selected by the planner from the candidates is y_2, and the final trajectory optimized by the controller is y_3. We choose two different cost functions $C(u)$ to evaluate the overall system. The cost described in Equation (2.1) can serve not only as a planner internal cost but also as a way to assess output trajectories by their holistic driving behavior. We also compute a second cost function, the safety cost, as the negative of the minimum distance between the ego agent and all other agents, across all steps in the planning time horizon.

We will consider three different planner designs. In one design, we will not incorporate either of the uncertainty measures that we have defined; this is the baseline planner as

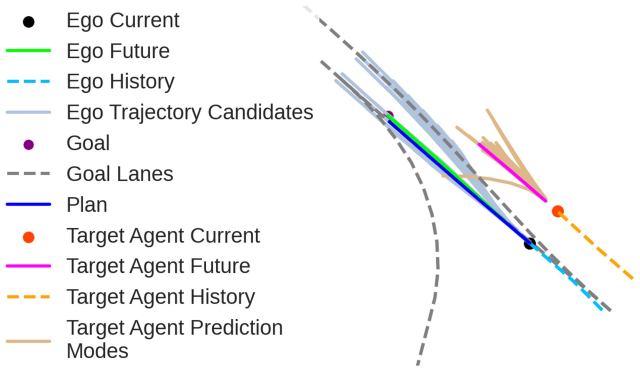


Figure 2.2: The Autonomous Driving system, with an example scene from the nuScenes dataset. The ego agent must navigate to its goal in the presence of other agents, of which only the nearest agent is shown.

described above. For our custom uncertainty-aware planner designs, we modify the planner's internal cost function to produce behaviors that are commonly desired in practice. In our agent-avoiding planner design, the planner reacts to higher uncertainty by becoming more conservative around the target agent:

$$C^{\text{int}}_{\text{avoid}}(\cdot; w) = w_1 e^{\alpha\sigma_1} C_{\text{coll}}(s, \hat{s}) + w_2 C_{\text{goal}}(s, g) + w_3 C_{l\perp}(s, m) + w_4 C_{l\angle}(s, m) + w_5 C_u(s) \quad (2.2)$$

with the additional parameter α. In our lane-keeping design, the planner gives less trust to the quality of the collision cost, which uses $\hat{s}$, and prioritizes staying centered in the lane, in response to higher uncertainty. Lane-keeping is often considered the default safe driving behavior:

$$C^{\text{int}}_{\text{lane}}(\cdot; w) = \sigma_1(w_1 C_{\text{coll}}(s, \hat{s}) + w_2 C_{\text{goal}}(s, g)) + \frac{1}{1-\sigma_1}(w_3 C_{l\perp}(s, m) + w_4 C_{l\angle}(s, m) + w_5 C_u(s)) \quad (2.3)$$

2.2 Runway Incursion Detection System

Another real-world system that we analyze is a runway incursion detection system for an aircraft that is about to land. The goal of the system is to produce advisory information to a pilot to help determine whether to land the aircraft or not, depending on whether an object on the ground, such as a vehicle, is likely to be intruding on the runway. We study this system with the help of a Boeing aircraft equipped with cameras. We also use a proprietary Boeing dataset that contains different landing scenarios with possible ground vehicle intrusions. This system is represented in Figure 2.3.

This is a 2-module system comprised of a detector (f_1) and a tracker (f_2). Given images of the runway, the detector produces bounding-box detections of potential objects on or near



Figure 2.3: The Runway Incursion Detection system, with an example image that might be input to the detector on the aircraft. From this aerial view of the runway, the aircraft must determine if there exist ground vehicles which could cause a runway incursion. Red circles correspond to detections. Yellow circles correspond to "probationary" tracks that are not fully initiated yet. Blue circles correspond to fully initiated tracks. The numbers correspond to detection confidence scores at that instant in time.

the runway. Since the images are taken from the air from potentially far away, there can be a high amount of uncertainty about whether the detected objects truly exist or not. The detector is a fully convolutional one-stage neural network model [147]. It is trained to also produce a confidence score between 0 and 1 per detection, where a score closer to 1 indicates a higher confidence that the detected object exists. We treat these confidence scores as the uncertainty measure from the detector.

The function of the tracker is then to associate objects to their detections over time and track their positions using a Kalman filtering-based approach. We design the tracker to be uncertainty-aware by incorporating the detector's confidence scores into a computation of the probability of an object's existence. These probabilities are used to determine when to initiate and terminate a track, under Wald's sequential probability ratio test (SPRT) [156].

Similar to the Autonomous Driving system, we define our relevant system quantities using sequences and histories of observations to meet our time-invariance assumption. x is defined to be the set of all camera images produced over the course of a track's existence as well as the overall environmental scenario. y_1 is the sequence of bounding-box detections corresponding to a single object track, and σ_1 is the sequence of confidence scores corresponding to the track. Finally, y_2 is the predicted probability of the track's existence as produced by the tracker. Using this continuous-valued probability as the output of the pipeline, as opposed to binary values indicating, for example, the presence of an incursion or not, allows us to have a more expressive evaluation cost function. We set this cost as the absolute difference between y_2 and the true probability of existence. Since the true probability of existence is an ill-defined quantity, we estimate it by dividing the number of predicted detections in a track that correspond to a ground-truth detection by the total number of ground-truth detections

in the track.

We will consider four different tracker designs. In one design, we will not incorporate uncertainty measures at all. Instead, the track initiation will be performed by simply checking if detections are being received for a high enough percentage of camera frames. Track termination will be performed by checking if detections have not been received for a certain amount of frames. We will also consider three uncertainty-aware tracker designs, denoted by "Naive", "Regression", and "Histogram". These were produced as part of different system design iterations and are differentiated by how they compute the likelihood ratio in the SPRT. The Naive tracker computes the likelihood ratio as the detector confidence divided by 1 minus the confidence. The Regression and Histogram trackers, however, more intelligently use the confidence scores by computing the mean and variance of the confidence scores over a rolling window. The Regression tracker then uses the mean and variance as features in simple autoregressive linear models for computing the numerator and denominator of the likelihood ratio. The Histogram tracker, meanwhile, uses prior data to construct histograms offline representing the conditional distributions in the likelihood ratio. Online, it uses the histograms to look up the probability corresponding to the rolling window's mean and variance.

Chapter 3

Measuring System Robustness

This chapter is based on work with Haotian Shen, Anish Muthali, Marco Pavone, Dragos Margineantu, Peter Karkus, Boris Ivanovic, and Claire J. Tomlin in [40] and [41].

3.1 Introduction

We desire to understand uncertainty-aware systems by examining their placement in the broader system design optimization landscape. Performance objectives often form an axis in this multi-objective space. In this work, we argue that uncertainty-awareness arises from considering a *robustness* objective in addition to that of performance. We thus propose an input-output robustness metric that can form this second objective. In essence, the robustness metric allows us to evaluate the impact of uncertainty quantification and help a designer to choose between alternative system designs for an application.

In producing a new design objective to *justify* why uncertainty quantification is useful, we take inspiration from [115], which derives the classic layered architecture in control systems from an optimization problem. Our robustness metric involves a search over perturbations or errors to the input to the system such that the system output has low cost. This problem is similar to that explored by [5], and the tools we use for the level-set estimation that this problem entails are also explored by prior works such as [67, 76, 84, 88]. Our work differs from these as follows: (1) We explicitly use our sub-level set to produce a *measurement* of robustness, and connect that measurement with choices of uncertainty-aware system designs, and (2) We examine how different system designs trade off between performance and this robustness metric, thus connecting uncertainty quantification with system design objectives. An overview of our method is shown in Figure 3.1.

3.2 Problem Statement

As mentioned in the previous chapter, we are interested in analyzing modular autonomous system pipelines involving components connected in sequence. Each module is a function f_i,

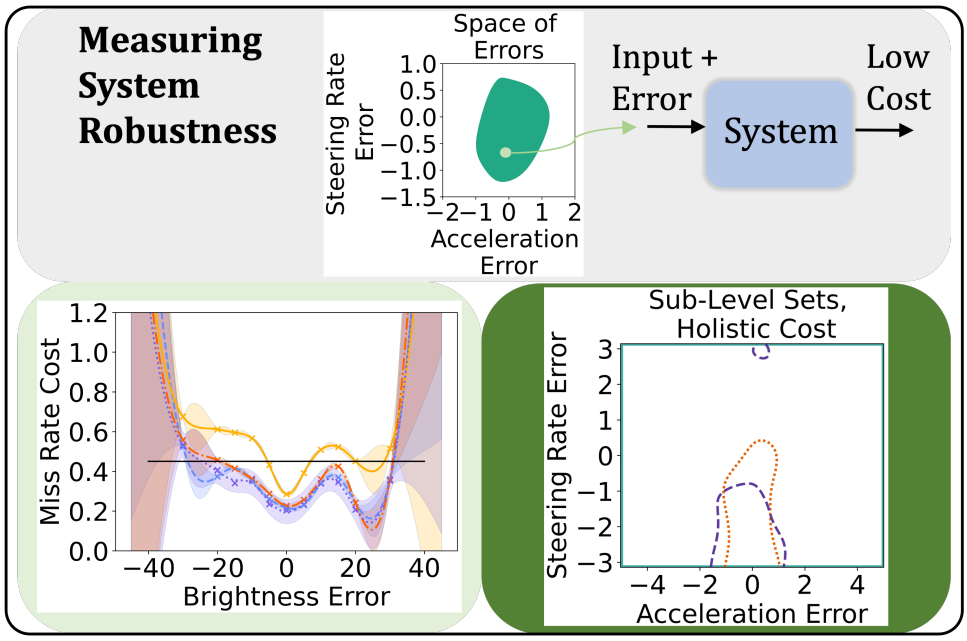


Figure 3.1: We propose to measure a system's input-output robustness by searching for all input errors such that the system output incurs an evaluation cost below a threshold. In the example results shown, we display functions and level sets that are the solution to the search problem. Different colors and line styles indicate different system designs. The light green color indicates evaluation on the Runway Incursion Detection system and the dark green color indicates evaluation on the Autonomous Driving system.

where $i \in \{1, ..., n\}$ indexes the module for n modules. The input to the entire system is the state $x \in \mathcal{X}$, which is a random variable drawn from a distribution $\mathcal{D}$. The output of the system is the control command $u \in \mathcal{U}$. Let us define the action of the system as a whole as $f := f_n \circ f_{n-1} \circ ... \circ f_1$. We also have access to an evaluation cost function $C : \mathcal{U} \to \mathbb{R}$.

A robust system is one that is performant even under perturbations to the input distribution $\mathcal{D}$. In our definition, this occurs when each x sampled from $\mathcal{D}$ is shifted to $p_\epsilon(x)$, where p is a known function with parameter ϵ. This shift or perturbation can arise from sensor errors, adversarial interference, or a disturbance in the environment. If x belongs to a high-dimensional space, $p_\epsilon(x)$ can represent a change to any of its semantic attributes. We thus formalize the notion of good performance under an input perturbation as the condition

$$\mathbb{E}_{x \sim \mathcal{D}}[(C \circ f)(p_\epsilon(x))] < c \tag{3.1}$$

for a designer-chosen threshold c. We briefly note here that although our formulation involves an expected cost value, other statistics can also be considered. Doing so may require a different methodology for calculating the final robustness metric, but would contain the

same ideas that we describe.

We would now like to search for *all* possible ϵ values such that the condition (3.1) is satisfied. Intuitively, if a system satisfies the condition for all ϵ values in a set with larger volume, it is performant under more distribution shift "amounts" and so is more robust. Formally, we need to find the set $\mathcal{S}$, where

$$\mathcal{S} := \{\epsilon \mid \mathbb{E}_{x\sim\mathcal{D}}[(C \circ f)(p_\epsilon(x))] < c\} \tag{3.2}$$

This is the set of input perturbations under which the system is still performant on average. Since a larger set corresponds to a more robust system, our final robustness metric is therefore the size of $\mathcal{S}$.

3.3 Method

In order to compute the size of $\mathcal{S}$, we observe that doing so essentially amounts to a sub-level set estimation problem for the function $\mathbb{E}_{x\sim\mathcal{D}}[(C \circ f)(p_\epsilon(x))]$ with argument ϵ. Since we assume that f is a black box, we perform this estimation via sampling. That is, we sample ϵ values, and for each sample, we use samples of x to estimate the expected value. Similar to prior work [67, 76, 84] based on the principles of Bayesian optimization [60], we fit a Gaussian Process (GP) [131] to the samples. This gives us an estimate of the function $\mathbb{E}_{x\sim\mathcal{D}}[(C \circ f)(p_\epsilon(x))]$. The fact that a GP models the stochasticity inherent in producing each sample allows for a probabilistic approach. In particular, we can take advantage of an acquisition function to determine where to sample in the space of ϵ and thus potentially reduce the computational complexity of the problem. In this work, we rely on the Maximum Improvement in Level-Set Estimation (MILE) acquisition function, introduced by Zanette et al [168]. Overall, this is one technique for performing sub-level set estimation, and other techniques may also be considered, especially if the statistic in the calculation is not the expected value, as was noted in the previous section.

Let us define $F(\epsilon)$ as $\mathbb{E}_{x\sim\mathcal{D}}[(C \circ f)(p_\epsilon(x))]$. Then, we can define $\hat{F}(\epsilon) = \frac{1}{N}\sum_{j=1}^{N}(C \circ f)(p_\epsilon(x_j))$ as the sample mean for N samples. Following one of the methodologies described by Katz et al. [88], we aim to find all ϵ such that

$$P(F(\epsilon) < c) > \gamma \tag{3.3}$$

We model $F(\epsilon)$ as a Gaussian distributed random variable with mean $\hat{F}(\epsilon)$. We then fit the Gaussian Process using samples $\{\hat{F}(\epsilon_1), ...\hat{F}(\epsilon_M)\}$ for M samples. Let $\mu_{GP}(\epsilon)$ and $\sigma_{GP}(\epsilon)$ be the resulting posterior mean and standard deviation functions. Then, we include a new point ϵ' into the sub-level set if $\mu_{GP}(\epsilon') + \beta\sigma_{GP}(\epsilon') < c$, where β is set such that a γ amount of probability mass is included for a Gaussian distribution. We find the size of $\mathcal{S}$ by gridding the space of ϵ and then counting the number of grid points that satisfy the above condition. We divide this number by the total number of grid points.

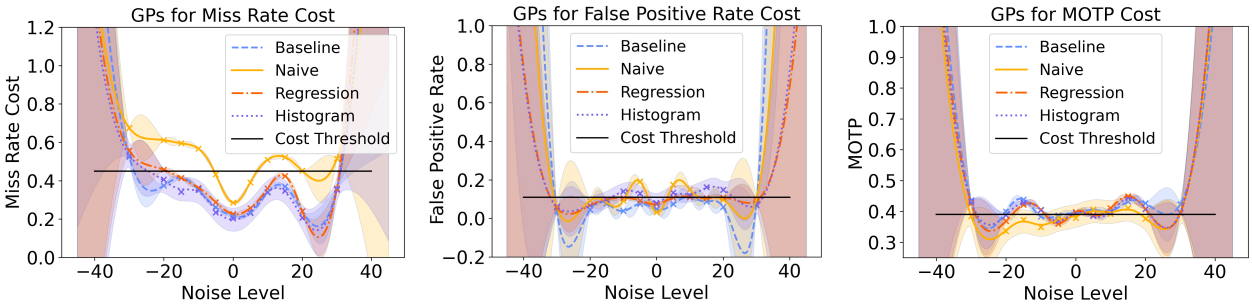


Figure 3.2: (Runway Incursion Detection System Case Study) We plot the fitted Gaussian Processes representing $F(\epsilon)$ for three different cost functions: the miss rate (left), false positive rate (middle), and MOTP (right). Thus, here "noise level" refers to the ϵ brightness perturbation value. Computing the sub-level sets of these functions yields a one-dimensional interval, which we then compute the size of by counting grid points.

Remark 1 [Computational Complexity]: The majority of the computation time of the technique we have described lies in producing estimates of $\mathbb{E}_{x\sim\mathcal{D}}[(C \circ f)(p_\epsilon(x))]$ for each ϵ sample. Thus, producing all of the samples for which we fit the GP takes time linear in M, N, and the simulation time for running f on each x. Again, this induces a tradeoff dictating the magnitude of these quantities. In future work, we would like to explore how system knowledge might aid in reducing this computational complexity. Once the GP is fit, some additional time, linear in the amount of grid points in the space of ϵ, is taken to compute the size of $\mathcal{S}$.

3.4 Results on Case Studies

For each of our Case Studies, we follow certain steps in our analysis: (1) We first define $p_\epsilon(x)$, (2) We enumerate the different system designs that we would like to compute the robustness metric for, and (3) We compute the sub-level sets for each system design and examine each design in terms of both robustness and performance.

Runway Incursion Detection System

Recall that for this system, x represents images of the runway from the aircraft camera. Since for this portion of our analysis we are interested in having a comprehensive understanding of the system's input-output behavior, we modify our definitions of x and u slightly. We have x represent all images in a test condition, and u represent all tracks produced during a test condition. Since in our formulation we allow for the perturbation function $p_\epsilon(x)$ to capture any kind of semantic shift to the input, we can solve our sub-level set estimation problem in

Table 3.1: Sub-Level Set Sizes and Costs for Tracker Designs

System	Miss Rate		FP Rate		MOTP	
	Size of SL Set ↑	Cost ↓	Size of SL Set ↑	Cost ↓	Size of SL Set ↑	Cost ↓
Baseline	**0.927**	0.217	**0.798**	0.048	0.0	0.395
Naive	0.218	0.286	0.0	**0.032**	**0.315**	**0.379**
Regression	0.774	0.227	0.556	0.075	0.218	0.398
Histogram	0.823	**0.204**	0.169	0.079	0.185	0.395

a potentially low-dimensional space. In this Case Study, we set $p_\epsilon(x)$ to represent a constant pixel-wise addition to each image in x by amount ϵ, capped at the values of 0 and 255 for the RGB images. This essentially changes the *brightness* of the image; positive ϵ values increase the image brightness while negative ϵ values make the image darker.

The system designs that we will compare will all include the same detector module, but with different tracker modules. We will compare the Naive, Regression, and Histogram trackers, as well as the tracker that does not use the uncertainty measure from the detector at all. We use the Miss Rate, False Positive Rate, and Multiple Object Tracking Precision (MOTP) as our three cost functions, which come from the tracking literature [18]. The miss rate and false positive rate measure how often a track does not note the existence of an object and maintains an object that does not exist, respectively, with respect to the number of ground truth objects. The MOTP measures how accurately the tracker maintains object positions over time with respect to ground-truth positions.

Figure 3.2 shows the Gaussian Processes that we fit to represent $F(\epsilon)$ for each cost function. For each ϵ sample, we sample test conditions 15 times, run all four systems for those samples, and compute the average cost per system. In terms of computation, although each test condition lasts less than five minutes, it is expensive to compute $F(\epsilon)$ due to overhead in running the system in ROS. We take 11 samples of ϵ in total. Since ϵ is only 1-dimensional in this case, we do not find it necessary to use the MILE acquisition function to achieve appropriate coverage of the space. The cost threshold c is an analysis choice and is set to 0.45 for the miss rate cost, 0.11 for the false positive rate cost, and 0.39 for the MOTP cost. We set γ from Equation (3.3) to 0.9. The GP priors have means that are specific quadratic functions and covariances specified by Matern52 kernel functions. To calculate the sizes of the level sets, we grid the ϵ axis in increments of 0.5.

In Table 3.1, we display the sizes of the sub-level sets of $F(\epsilon)$ in terms of the percent of grid points included in the sets, as well as the performance of each system design as measured by the cost value itself when $\epsilon = 0$. In this way, we address our original intent of understanding how uncertainty-awareness affects a system's placement in the system design optimization landscape. Clearly, a design may produce a good performance but lack in robustness, and vice versa.

Examining our results, we see that an uncertainty-aware design using the Histogram tracker performs better than the baseline no-uncertainty system in terms of miss rate performance. However, it as well as the Regression tracker design perform worse in terms of the false positive rate performance. This is expected since the uncertainty-aware trackers are designed to be more conservative, maintaining tracks for longer periods of time and not requiring very high confidence score-detections to initiate a track in the first place. This conservatism produces a tradeoff in terms of miss rate and false positive rate. We also see that the Naive tracker performs the best in terms of MOTP. This is because it only uses confidence scores at each timestep to determine whether to initiate or terminate a track, instead of using a history of confidence scores. As a result, it often produces very short tracks, terminating a track once a confidence score drops instead of maintaining it for long periods. We believe that this allows it to be more precise in estimating an object's location, as the tracked object's location matches more closely with the detections.

In terms of our robustness results, we see that interestingly the baseline design has the highest sub-level set size for both the miss rate and false positive rate cost functions. However, the uncertainty-aware designs have a higher robustness for the MOTP cost function. One reason for this may be that as we add perturbations to the input image, the detector may be producing worse detections, for example by dropping detections or detecting non-objects, in a way that impacts trackers similarly. Indeed, the data points for the Baseline and Histogram tracker system GPs tend to be close, at least for the miss rate cost. If the detector produces the wrong detections, a tracker, even an uncertainty-aware one, may not be able to do much to "recover" and have a good performance. However, in general we do see the presence of tradeoffs that we predicted – while the Baseline system has advantages in some ways, it does not have a good MOTP robustness or performance. Similarly, the Regression tracker system performs reasonably well in terms of miss rate and false positive rate robustness but poorly in terms of MOTP performance.

Autonomous Driving System

For this system, x represents the control history of the target agent for the trajectory predictor. We define $p_\epsilon(x)$ as representing a perturbation to the acceleration and steering rate of the final, most recent step of the history. That is, ϵ is two-dimensional, consisting of $\epsilon_{\text{accel}} \in [-5, 5]$ and $\epsilon_\omega \in [-\pi, \pi]$. We add ϵ to the last control action and re-compute the history trajectory according to the agent dynamics.

The system designs that we will compare will involve different combinations of uncertainty heuristics and uncertainty-aware planner designs: $(\sigma_{ML}, C^{\text{int}}_{\text{avoid}})$, $(\sigma_{ML}, C^{\text{int}}_{\text{lane}})$, $(\sigma_K, C^{\text{int}}_{\text{avoid}})$, and $(\sigma_K, C^{\text{int}}_{\text{lane}})$. We will also involve the baseline system in our comparison. For our cost functions, we use the same holistic and safety cost functions defined in the previous chapter. We take costs *relative* to that produced by a system with access to the ground truth target agent future. This is because autonomous driving offers a system a wide variety of possible scenarios in which it is difficult to assess performance using absolute cost. For example, a

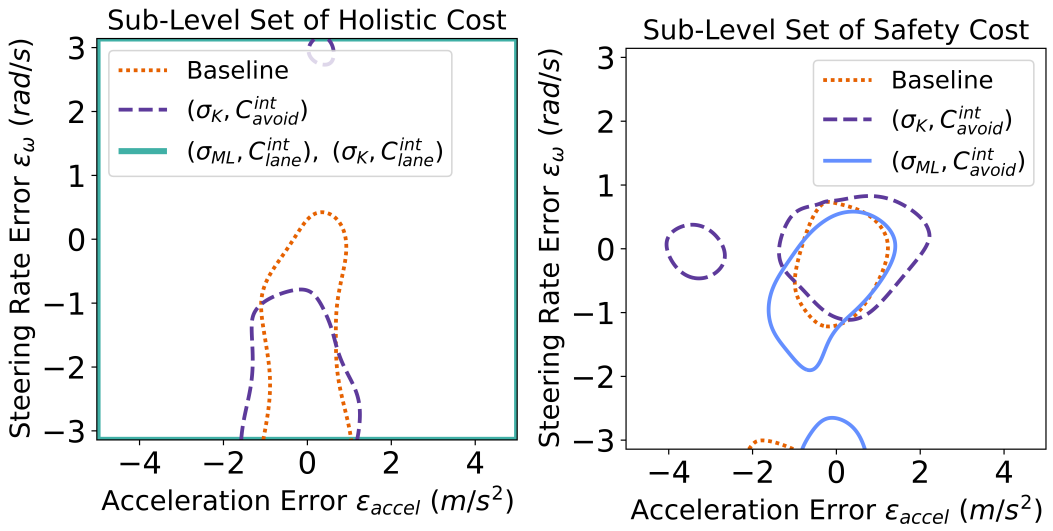


Figure 3.3: (Autonomous Driving System Case Study) We show the sub-level sets of $F(\epsilon)$ for the holistic (left) and safety (right) cost functions. For the holistic cost, $c = 0.05$, and for the safety cost, $c = 0.02$. We omit sub-level sets of size 0. Some sub-level sets occupy the whole space. Part of the sub-level set for the Baseline system for the safety cost is also shown in Figure 3.1.

cost may be high because the scenario is inherently difficult to navigate, or inherently low because perhaps all non-ego agents are very far away and do not impact the ego agent.

For this Case Study, we fit the GPs that represent $F(\epsilon)$ over the two-dimensional space of ϵ. We first take 15 random samples of ϵ for an initial GP fitting and then use the MILE acquisition function to sample 20 additional times. The cost threshold c is set to 0.05 for the holistic cost and 0.02 for the safety cost. We set γ from Equation (3.3) to 0.9. In Figure 3.3, we display the obtained level sets. For each ϵ sample, we evaluate each system on a subset of the nuScenes dataset that is unseen by Trajectron++ at training time and compute the average cost. This amounts to 12,452 data points. Each $F(\epsilon)$ evaluation takes on the order of 15 minutes. The GP priors have zero means and covariances specified by Matern52 kernel functions. To calculate the sizes of the level sets, we grid the acceleration error axis in increments of 0.02 and the steering rate error axis in increments of 0.015.

In Table 3.2, we compare the performance and robustness of the systems. We find that for both the holistic and safety costs, there exists a design involving one of our uncertainty measures and uncertainty-aware planners that is better than the baseline in terms of both performance and robustness. As one might expect, using the agent-avoiding planner is best for safety cost performance and robustness. We also point out the particular efficacy of using the lane-keeping planner in terms of the holistic cost performance and robustness. This shows the promise of lane-keeping behavior as a strategy in the presence of input errors.

Table 3.2: Sub-Level Set Sizes and Costs for AV Designs

System	Holistic Cost		Safety Cost	
	Size of Sub-Level Set ↑	Cost $\times 10^{-2}$ ↓	Size of Sub-Level Set ↑	Cost $\times 10^{-2}$ ↓
GT Prediction	—	0	—	0
Baseline	0.092	1.44	0.036	0.46
$(\sigma_{ML}, C^{\text{int}}_{\text{avoid}})$	0.0	**-0.76**	0.036	0.38
$(\sigma_{ML}, C^{\text{int}}_{\text{lane}})$	**1.0**	-0.49	0.0	7.77
$(\sigma_{K}, C^{\text{int}}_{\text{avoid}})$	0.084	2.90	**0.073**	**-2.63**
$(\sigma_{K}, C^{\text{int}}_{\text{lane}})$	**1.0**	-0.19	0.0	8.28

The coupling between the uncertainty measure and the decision-making-under-uncertainty algorithm, however, proves to be very important. For example, the agent-avoiding planner has a good holistic cost when σ_{ML} is used, but a poor cost when σ_K is used.

We can also more deeply analyze the sub-level sets in Figure 3.3 to yield useful insights about the system. For instance, the holistic cost level sets tend to have volume when $\epsilon_\omega < 0$ but not when $\epsilon_\omega > 0$. We found two reasons for this. For one, we found that our trajectory predictor Trajectron++ performs better when $\epsilon_\omega < 0$ than when $\epsilon_\omega > 0$, perhaps due to biases in the training data. Secondly, when the steering rate error is negative the target agent's prediction is often such that the ego agent's plan is less likely to intersect with it and can more directly reach the goal.

Thus, our method serves to not only help a designer measure a design's robustness, but also produce a better understanding of the system. It is not only a tool for comparing designs but also for helping to diagnose vulnerabilities and weaknesses in potential future operating conditions. This can additionally be helpful from a system safety validation standpoint: our analysis provides a designer with the set of operating conditions for which a system performs well in addition to the conditions for which it does not. Subsequent data-gathering efforts can then be concentrated on the regimes of lower performance.

Acknowledgments

This material is based upon work supported by the DARPA Assured Autonomy Program, the NASA ULI on Safe Aviation Autonomy, and the NSF GRFP under Grant No. 2146752. Any opinions, findings, and conclusions or recommendations expressed in this material are ours and do not necessarily reflect the views of any aforementioned organizations. Additionally, we would like to thank Brandon Schwiesow, Jose Medina, Zachary Tane, Blake Edwards, James Paunicka, and the rest of the team at Boeing for their help and contributions.

Chapter 4

Generating Specifications on Module Uncertainty

This chapter is based on work with Haotian Shen, Anish Muthali, Marco Pavone, Dragos Margineantu, Peter Karkus, Boris Ivanovic, and Claire J. Tomlin in [40] and [41].

4.1 Introduction

As autonomous systems become more pervasive in industry and in daily life, we see simultaneously the trends towards incorporating machine learning into these systems to give them greater generality and flexibility as well as the need for these systems to be safe and reliable. Many of these systems are modularized for interpretability and ease of engineering design. For example, a classic robotics system architecture may involve perception and localization modules, whose outputs are then passed on to planning and control modules. Some modules, such as the perception module, may involve learning, while others, such as the control module, may not. In recent times, producing assurances on system operation in the presence of these machine learning components has been an especial challenge.

In the machine learning literature, a common way to provide more trust in learned models is to perform uncertainty quantification [25, 109]. Techniques such as anomaly detection [17], out-of-distribution (OOD) detection [8, 72, 101, 108, 139], and epistemic and aleatoric uncertainty quantification [63, 89, 94, 96, 130], exist to help a model "know that it does not know". However, the uncertainty measures produced by these techniques can often be unintuitive [31, 166] and require calibration [9, 49, 118]. Additionally, there is no consensus on how to define some of the terminology for these techniques, such as "out-of-distribution", and it is not often clear which technique one should use [129]. Thus, the uncertainty quantification technique and the type of decision-making-under-uncertainty algorithm used by a system tend to be heuristic design choices.

While much of the work in the machine learning literature regarding uncertainty quantification considers standalone modules, in the current paradigm, learned modules are part

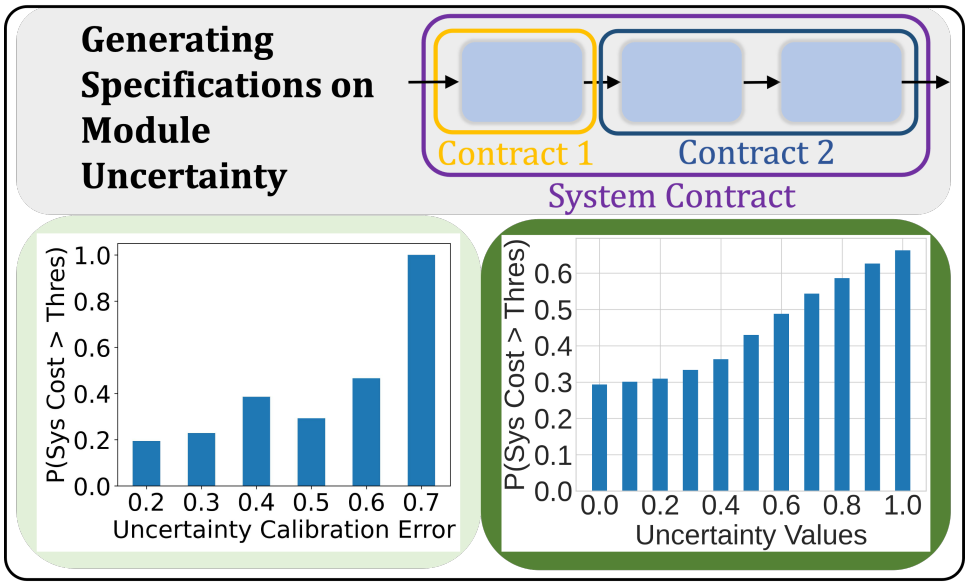


Figure 4.1: We propose that generating a specification on a module's uncertainty measure will help us to apply our system-level perspective. Using assume-guarantee contract theory, we treat this as a problem of finding the contract of an upstream module of interest (Contract 1) given the system contract and the contract of the downstream decision-making modules (Contract 2). We show some example results in which we use data to generate histograms that help to characterize the Contract 2. The light green color indicates evaluation on the Runway Incursion Detection system and the dark green color indicates evaluation on the Autonomous Driving system.

of a larger decision-making system. As such, we argue that uncertainty measures are given meaning not by themselves, but in the context of how they are used by the system in the operating environment. That is, the uncertainty measure of a module should be understood *relative* to the efficacy of the decision-making-under-uncertainty algorithm employed by the downstream modules.

Here, we present an analysis technique aimed at providing a better understanding of uncertainty quantification from a system-level perspective. We consider the problem of generating a specification on a module's uncertainty measure, given a system-level specification. We use assume-guarantee contract theory to formally define the problem. We then write our specification as a particular inequality that connects three quantities: the relationship between a module's uncertainty measure and its error, how that uncertainty measure is used downstream, and the overall system performance. Our framework allows us to simultaneously identify the characteristics of the uncertainty measure and the downstream decision-maker that will lead to good overall system performance.

This analysis is applied on two realistic, complex, and industry-grade autonomous sys-

tems: an autonomous driving system and an aircraft runway incursion detection system. These systems are described in detail in Chapter 2. Given that system-level design questions such as ours can be computationally intensive to answer, we also include discussion of the computational complexity of our methods and the tradeoffs that can be necessary to make our analyses feasible. An overview of our method is shown in Figure 4.1.

4.2 Related Work

Modules and Uncertainty Measures in the Context of the System

The fundamental concept behind our work is that of taking a system-level perspective on the characteristics of individual modules. This philosophy is drawn from several recent works. In [82] and [122], module errors are considered in terms of their downstream impact. Doing so can help to re-design modules in a system-level risk-aware way, such as in [38]. Works such as [165] and [171] also design certain modules in an autonomy pipeline by explicitly taking into account how they interact with other modules.

Various works also connect module uncertainty measures with system-level objectives. [10] and [54] aim to produce an uncertainty measure and OOD detector, respectively, according to the desideratum that they should help to predict downstream performance. In contrast, we try to shift away from defining a new or better uncertainty measure and towards a deeper understanding of the role of uncertainty quantification in the overall system. In the first analysis in our work, we use assume-guarantee contract theory to define what *property* we care about with respect to a module's uncertainty measure. Thus, we essentially define desiderata for an uncertainty measure in a more rigorous way.

Similar to our work, several other works connect the concept of *calibration* with system-level perspectives on uncertainty measures. For example, [97] calibrates the decision outputs of the autonomy pipeline directly with respect to risk instead of calibrating upstream uncertainty measures. The works [90] and [167] also recognize the need for calibrated prediction sets produced at the module level. Both works ask how to produce prediction sets representing a learned model's uncertainty that are *optimal* with respect to a downstream decision-making cost, for example using end-to-end learning.

In our work, we present a more general perspective, in which we treat *any* intelligent downstream decision-maker as performing calibration of the upstream uncertainty measure with respect to system-level costs, which is what gives meaning to an uncertainty measure in the first place. Indeed, we can quantitatively understand a decision-maker's efficacy as the degree to which this occurs. Additionally, we allow for more heuristic approaches to uncertainty representation and decision-making, as can be more commonly implemented in real-world systems. While works such as [90] can make theoretical statements about optimality, they only consider uncertainty representations in the form of prediction sets and decision-making in the form of robust optimization. Thus, our work has a more practical orientation.

Specification Generation and Assume-Guarantee Contract Theory

In this work, we aim to produce a module-level specification given a system-level specification. One perspective on this problem is given by Katz et al. [88], in which the system-level specification is that the probability of a system-level failure should be below a threshold. Finding all module behaviors such that this is true is represented and solved as a sub-level set estimation problem, with all desired module behaviors therefore lying within the set.

One can also turn to the formal methods literature more broadly to consider the specification generation problem. In this work, we use assume-guarantee contract theory [77, 124, 133], which allows us to formally define module properties and interconnections. In particular, we draw upon the concept of the quotient contract as introduced by Incer et al. [78, 79]. We extend [78] by producing a novel quotient contract formulation involving module uncertainty measures. Our formulation is additionally novel due to its particular *probabilistic* form.

4.3 Definitions and Assumptions

Our key focus is on analyzing modularized autonomy pipelines involving components connected in sequence. Such pipelines are usually designed to first involve estimation-related modules, such as those performing perception or filtering, followed by decision-making-related modules, such as those performing high-level planning and low-level tracking and control. We define "system" as this modularized pipeline, as opposed to the controlled object that uses the controls to evolve the state.

Mathematically, we write each module as a function f_i, where $i \in \{1, ..., n\}$ indexes the module for n modules. The input to the system is the state $x \in \mathcal{X}$, which is a random variable drawn from a distribution $\mathcal{D}$. The output of the system is the control command $u \in \mathcal{U}$. Each module produces a nominal output $y_i \in \mathcal{Y}_i$ and also may be associated with an uncertainty quantification technique that produces an uncertainty measure $\sigma_i \in \Xi_i$. Additionally, we can compute a measure of error for each module given y_i and a ground truth value $y_i^* \in \mathcal{Y}_i$. We denote module i's error value as $e_i \in \mathcal{E}_i$. Finally, we have access to an evaluation cost function $C : \mathcal{U} \to \mathbb{R}$. We point out that since x is a random variable, every y_i, e_i, and σ_i is as well, along with u and $C(u)$. Though we place no restrictions on $\mathcal{X}, \mathcal{U}$, and all $\mathcal{Y}_i, \mathcal{E}_i$, and Ξ_i, for many systems these are often spaces of vectors, matrices, distributions, or sets. For example, each σ_i may be a vector, distribution, or simply a scalar.

Let Ω_i be the space of sets of the form $\{a, b\}, a \in \mathcal{Y}_i, b \in \Xi_i$ for $i \in \{1, ..., n\}$. Formally, we write that $\mathcal{U} = \mathcal{Y}_n$ and that $f_1 : \mathcal{X} \to \Omega_i$ and $f_i : \Omega_{i-1} \to \Omega_i \ \forall i \in \{2, ..., n\}$. Given a module $f_{i'}$, we denote the "decision-making-under-uncertainty algorithm" relative to this module as the function composition $f_n \circ ... \circ f_{i'+1}$. In the case that a module does not involve uncertainty quantification, the associated σ simply takes on a 'null' value.

We will also find it necessary to define the discrete random variable z that represents the type of environmental scenario that the system encounters. Since z is discrete and

environmental information is often present in x, z can be thought of as a compression of x.

A key assumption in this work is that each module is deterministic and time-invariant given these input-output definitions. This means that each forward pass of the pipeline corresponds to an open-loop decision-making output. As discussed in our Case Studies in Chapter 2, this assumption may require definitions of x, u, and $C(u)$ that capture all the information needed to run the system and evaluate its performance. For example, x can involve a history of observations and u can be a sequence of actions.

4.4 Problem Statement

As we consider contextualizing uncertainty measures within the overall system, we require a way to reason about how the value of an uncertainty measure can lead to either desirable or undesirable system outcomes. For this reason, we choose to study the question of how to generate a module-level specification on uncertainty measure values given a system-level specification. In our setting, we consider a *probabilistic* system-level specification:

$$P(C(u) > c) < \alpha \tag{4.1}$$

for designer-chosen thresholds c and α. This specification thus describes the set of allowable distributions over $C(u)$.

We would like to produce a specification to constrain the values of a target module's uncertainty measure. However, this problem as written is not clearly defined. In the same way that the system specification is a constraint on a distribution, what property should the module-level specification constrain? Here, a *property* is defined as a set of behaviors [77]. However, an uncertainty measure by itself is not a module property but the output of an uncertainty quantification technique, and has many characteristics that we may want to constrain.

We turn to assume-guarantee contract theory to define our properties. An assume-guarantee contract (A, G) is a pair of properties, the *assumptions* A and *guarantees* G. A module or a system implements a contract if it satisfies the guarantees given that it operates in any environment that satisfies the assumptions [77]. The problem of obtaining a module-level specification given a system-level specification is very related to that of taking a *quotient contract* [79]. That is, given the system-level contract and the contract of the remaining modules of the system, one can compute the contract of the module of interest. If the contract of the remaining system modules is not already known, it must be obtained, for example using data-driven characterizations. In the next section, we use this framework to write down specific criteria that a designer must consider when defining uncertainty-related properties. We then proceed to solve a quotient contract problem.

4.5 Generating Specifications on Uncertainty Measures

Throughout this section, we will use the Autonomous Driving system with the agent-avoiding planner (using the internal cost function given by Eq. (2.2)) as our running example.

Running Example: *We focus on σ_1, which is a scalar value extracted from the output of the trajectory predictor module. The output of the system, u, is the final ego trajectory produced by the system, and the system cost function, $C(u)$, encapsulates desired driving behavior, such as holistically good driving or safe driving. In our problem statement, we desire a way to use the system-level specification (4.1) to produce a module-level specification relating to the predictor module's uncertainty measure.*

At a high level, we aim to address the problem in Section 4.4 by first defining the uncertainty-aware module property involved in our specification, and then writing the specification as a particular inequality. We do this using tools from assume-guarantee contract theory ("Assume-Guarantee Contract Theory for Uncertainty"). Our specification involves a probability distribution that characterizes the behavior of the modules downstream of the target module. We need to estimate the probabilities in that distribution to have a working specification ("Estimating Constants in the Specification"). Then, we discuss how our specification provides insights into the efficacy of the decision-making-under-uncertainty algorithm and the desired relationship between the uncertainty measure and module error ("Discussion"). We finally perform evaluations on our Case Studies ("Results on Case Studies").

Assume-Guarantee Contract Theory for Uncertainty

Let the module of interest be module i. Then we refer to the contract implemented by modules 1 to i as (A_1, G_1) and the contract implemented by modules $i+1$ to n as (A_2, G_2). We write the contract implemented by the system as a whole as (A_{sys}, G_{sys}). We also write $A_i \in \mathcal{A}_i$ and $G_i \in \mathcal{G}_i$ for $i \in \{1, 2\}$ and $A_{sys} \in \mathcal{A}_{sys}$, $G_{sys} \in \mathcal{G}_{sys}$, where $\mathcal{A}_{sys}$, $\mathcal{G}_{sys}$, and all of the $\mathcal{A}_i$ and $\mathcal{G}_i$ are property spaces. While we may not have a formal expression for A_{sys}, it represents the set of conditions under which the system is designed to operate. G_{sys} is then the set of distributions over $C(u)$ such that the system-level specification (4.1) is satisfied. Since the composition of the n modules describes the system's behavior, we can also set A_1 equal to A_{sys} and G_2 equal to G_{sys}.

Running Example: *Let us refer to the contract implemented by the trajectory predictor as (A_1, G_1) and the contract implemented by the planner and controller combined as (A_2, G_2). When we take the quotient contract, we use (A_{sys}, G_{sys}) and (A_2, G_2) to compute (A_1, G_1). The desired module-level specification would be one that determines G_1. However, we do not know yet what $\mathcal{G}_1$ or $\mathcal{A}_2$ are.*

We can thus reframe our question as trying to determine the property space $\mathcal{G}_1$. We know that for a module i implementing contract (A_i, G_i) connected to a module $i+1$ implementing

contract (A_{i+1}, G_{i+1}), it is true that $G_i \supseteq A_{i+1}$ [169]. Intuitively, module $i+1$ cannot "request" more than module i can "provide". So, we write that $\mathcal{G}_1$ must abide by the following criteria:

Criteria for Choosing the G_1 Object:
(a) $\mathcal{G}_1$ should be related to the uncertainty measure σ_i in some way.
(b) It must allow for tractable and interpretable analysis.
(c) It must be such that $G_1 \supseteq A_2$ for all $G_1 \in \mathcal{G}_1$ and $A_2 \in \mathcal{A}_2$.

Given these criteria, the designer must choose $\mathcal{G}_1$ depending on the problem setting. In particular, Criterion (c) indicates that we must consider what factors the downstream modules from module i depend on for their performance. Those factors then help to dictate what $\mathcal{A}_2$ is. In our analyses, we have found that these factors are often related to the type of environmental scenario z and the relationship between the uncertainty measure σ_i and module error e_i.

In our particular perspective, we define A_2 as being a set of distributions $P(g_{enc}(\sigma_i, e_i)|z)$, where g_{enc} is a function that encodes σ_i and e_i in a way that captures their relationship. Thus, $\mathcal{A}_2$ is the space of sets of distributions of the form given. Now, we set G_1 equal to A_2 (and therefore $\mathcal{G}_1 = \mathcal{A}_2$).

Running Example: *As mentioned above, the performance of the planner and controller depends on the type of environmental scenario and the relationship between the trajectory predictor's uncertainty value and its error. Thus, we write A_2 as a set of distributions $P(\sigma_1, \mathbf{1}(e_1 < e_{thres})|z)$, and $A_2 = G_1$. So, $g_{enc}(\sigma_1, e_1)$ here outputs $\{\sigma_1, \mathbf{1}(e_1 < e_{thres})\}$ for an error threshold e_{thres}. z is defined as $\mathbf{1}(distance(ego, target) < d_{thres})$ for a distance threshold d_{thres}. Note that G_1 is still unknown; we have simply defined what kind of object it is. The specification that we will generate will delineate which distributions $P(\sigma_1, \mathbf{1}(e_1 < e_{thres})|z)$ belong in G_1.*

We now write the specification by first using the Law of Total Probability. We abbreviate $g_{enc}(\sigma_i, e_i)$ as l_i and write

$$P(C(u) > c) = \sum_z \sum_{l_i} P(C(u) > c|l_i, z)P(l_i|z)P(z) \tag{4.2}$$

We note that l_i is a continuous random variable. However, as we go forward we will rely on a discretization of l_i, thus justifying the use of the summation above. The reasons for this are related to the ease of producing and using the specification, which will be explained further in Section 4.5.

Now, we use Eq. (4.1) to incorporate the system-level specification:

$$\sum_z \sum_{l_i} P(C(u) > c|l_i, z)P(l_i|z)P(z) < \alpha \tag{4.3}$$

We see that if the $P(C(u) > c|l_i, z)$ and $P(z)$ values are known, this inequality produces a constraint on the values that $P(l_i|z)$, that is $P(g_{enc}(\sigma_i, e_i)|z)$, can take on. In other words,

Eq. (4.3) is the desired specification over $P(l_i|z)$, giving the set of distributions of that form such that $P(C(u) > c) < \alpha$. G_1 is therefore implicitly described by this inequality.

We connect back to the concept of the quotient contract by noting that the quantity $P(C(u) > c|l_i, z)$ describes how well the downstream decision-making-under-uncertainty modules perform given σ_i and e_i. That is, in Eq. (4.3) we are using both knowledge of the system-level contract (A_{sys}, G_{sys}) and of how the downstream modules relate A_2 to G_2 to compute G_1[1].

Running Example: *Eq. (4.3) provides a specification by determining values that the distribution* $P(\sigma_1, \mathbf{1}(e_1 < e_{thres})|z)$ *can take; that is, by dictating how much probability mass can be placed for different values of* σ_1 *and* $\mathbf{1}(e_1 < e_{thres})$ *given different values of* z*.* $P(z)$ *refers to the distribution that the designer can expect the environmental scenarios to follow with respect to the distance between the ego and target agent.* $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ *represents how the planner and controller react to the upstream trajectory predictor's uncertainty and error in terms of the final cost incurred.*

What remains is now to provide estimates of the $P(C(u) > c|l_i, z)$ and $P(z)$ values and point out how this aids in our analysis. This will be discussed in the next two sections. However, before we proceed we make the following remark.

Remark 1 [Choosing g_{enc} and z]: We point out the importance and nuances behind the choice of g_{enc} and z that the designer makes for analysis. Here, Criterion (b) plays a key role. While the specification (4.3) is true no matter how g_{enc} and z are defined, the usefulness of the empirical analyses that we perform is dependent on these variables being interpretable and relevant to system performance. For example, for interpretability we may desire that $C(u)$ not be independent of $g_{enc}(\sigma_i, e_i)$ and z. However, if one simply sets z equal to x and g_{enc} to the identity mapping, the specification (4.3) loses its meaning as constraining module properties. Thus, overall our criteria provide the designer with much flexibility to define variables, though with these caveats.

In our Case Study for the Runway Incursion Detection system in Section 4.5, we will show how g_{enc} can be defined as a *calibration error* function, and explain the implications. Indeed, in that case study we even remove dependencies on z. In this way, our formulation operates in as much generality as possible. Future work can explore additional ways to define these variables; for example, one can imagine defining z using learned embeddings of data, similar to [48].

Estimating Constants in the Specification

As mentioned in the previous section, a key part of our analysis lies in the estimation problem of finding the $P(C(u) > c|l_i, z)$ and $P(z)$ values. We treat l_i as a discrete random variable in this estimation for several reasons. Firstly, by doing so we avoid making assumptions on the

[1] On a technical note, since G_1 is a set of distributions, it is not actually a property but a *hyperproperty*, or a set of properties. However, the notion of the quotient can still apply by using the concept of *hypercontracts* as introduced by Incer et al. [77], and the intuitions behind our formulation stay the same.

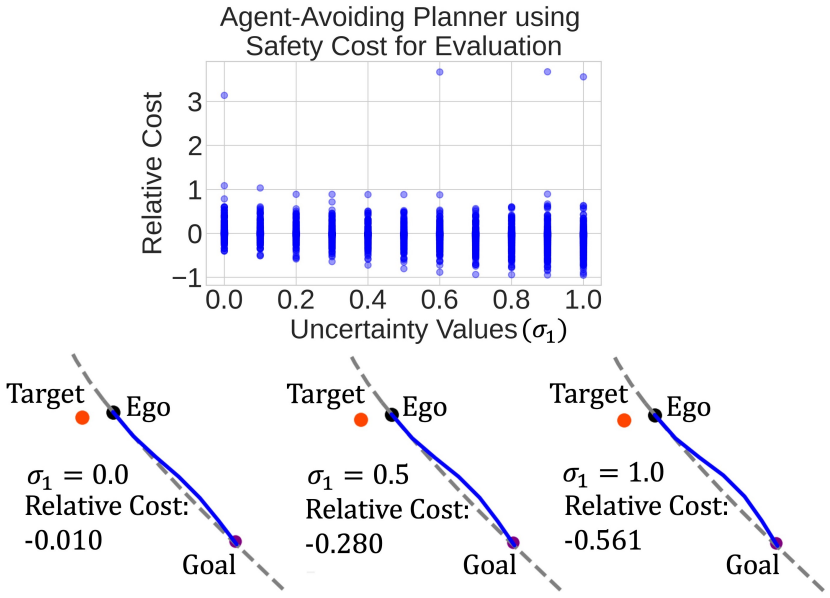


Figure 4.2: (Running Example) (Top) A plot showing the raw data used to estimate $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ for when the prediction error is below the threshold and the ego agent and target agent are close together. (Bottom) Plots showing one of the scenarios in the dataset with the associated plan for three different settings of the uncertainty value. Since this is the agent-avoiding planner, we indeed see that as the uncertainty value increases, the plan (dark blue solid line) takes a more conservative maneuver to reach the goal. The evaluation cost here is the safety cost.

form of these distributions, for example by parametrization, for performing the estimation itself. Secondly, we simplify the nature of the specification for the designer. In many real-world scenarios, a designer will check if their distribution $P(l_i|z)$ satisfies the specification by taking samples of the input x to the system and running the desired modules. In such cases, it is most natural to calculate the $P(l_i|z)$ values by performing counting per l_i bin. Also, if l_i is discrete the $P(C(u) > c|l_i, z)$ and $P(z)$ values have a nice interpretation of being weights to the $P(l_i|z)$ values in the specification. This has implications for better understanding the efficacy of the decision-making-under-uncertainty modules that are using the uncertainty measure from module i, as we will show.

We first address the issue of the $P(z)$ values. These are estimates of the relative frequency of the types of scenarios that the system will encounter. This can be produced by examining a representative dataset. At design time, these estimates serve to weigh the $P(C(u) > c|l_i, z)$ values, indicating how important it is for the decision-making-under-uncertainty modules to perform well for different situations.

For the estimation of the $P(C(u) > c|l_i, z)$ values, we similarly use a dataset of inputs to the system and environmental scenarios. We use this dataset to sample outputs from the

decision-making-under-uncertainty modules, given input y_i and σ_i values. The cost function can then be evaluated for every output. We discretize l_i by creating bins and rounding values into the bins. Then, for each bin value of l_i and each value of z, we can compute the probability that $C(u) > c$ using counting.

Running Example: *We estimate the* $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ *values, representing the action of the planner and controller, using the following method, though other methods are also possible. We first sort the dataset into scenarios in which the ego and target agent are close together and those in which they are far apart. For the trajectory predictor error, we use the ADE (average displacement error) between the predicted most likely trajectory of the target agent and the ground truth trajectory. For a threshold* e_{thres} *on this value, we consider data points for which the trajectory predictor produces predictions with error less than and also greater than this value. Then, we grid the space between* 0 *and* 1 *and select each grid point to be a* σ_1 *sample. For each data point, we run the planner and controller for each* σ_1 *value and for the fixed trajectory prediction. A sample plot showing* σ_1 *plotted against* $C(u)$ *is shown in Figure 4.2 for the safety cost for one selected value of* $\mathbf{1}(e_1 < e_{thres})$ *and* z*. Finally, we show plots in Figure 4.3 that give the final* $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ *values for all* $\mathbf{1}(e_1 < e_{thres})$ *and* z *value combinations. Here, for our cost we use the relative cost; that is, the cost of the trajectory produced by the system minus that produced by a system with access to the ground truth target agent future.*

One important observation from our approach is that our specification is data-driven and therefore approximate. As the data that is used for the probability estimations becomes more representative of the system and environmental scenarios, the accuracy of the approximation also increases. The burden is therefore on the designer to produce good estimates. Another note that we make is on the discretization strategy that we have employed. In practice, the l_i bin sizes should be small to produce a fine discretization of the continuous-valued l_i but also large enough to ensure that there are enough samples in each bin. In future work, we would like to perform more rigorous statistical analysis that takes into account these approximations in order to make our specification more robust. For instance, a distributionally robust specification can be developed, or a specification that accounts for the variance of each probability estimate.

Remark 2 [Computational Complexity]: Overall, the computational complexity of our technique comes from the estimation of the $P(C(u) > c|l_i, z)$ values, which requires a dataset, possibly a way to simulate the decision-making-under-uncertainty modules for different inputs, and an evaluation of the cost function for each data point. Thus, the computation time is linear in the size of the dataset and the runtime of the decision-making forward pass and cost evaluation. Given a fixed computational budget, this induces a tradeoff dictating how many data points can be processed.

Discussion

In this section, we discuss the implications that this specification generation process has on understanding uncertainty quantification and decision-making under uncertainty. Firstly,



Figure 4.3: (Running Example) For the agent-avoiding planner, we show the final $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values for (left) evaluation using the safety cost and (right) evaluation using the holistic cost. For each plot, we set the values of $\mathbf{1}(e_1 < e_{thres})$ and z and show how the probability values vary by the uncertainty value. The value of e_{thres} is 1.5, the value of c is 0.02, and a low ego-agent distance is defined as a distance less than 5 meters, while a high ego-agent distance is a distance greater than 5 meters. In general, these values can be chosen arbitrarily by the system designer. As the uncertainty value increases, the agent-avoiding planner keeps the ego agent further away from other agents, which can make the ego safer but worse at holistic driving.

the specification informs the system designer on the set of permissible distributions that govern a particular module's uncertainty measure such that the overall system performs well in probability. In our formulation, this distribution relates the uncertainty measure with module error in some way. So, the designer is informed of how much effort is needed to select or calibrate an uncertainty measure depending on how restrictive the specification is.

Secondly, the $P(C(u) > c|l_i, z)$ weights in the specification can be examined to yield additional insights on *how well* the uncertainty measure is used by the downstream modules. For example, if l_i encodes a kind of distance between the uncertainty measure and the module error, thus capturing how well the uncertainty measure correlates with error, then the $P(C(u) > c|l_i, z)$ weights show the downstream modules' sensitivity to that value. If the $P(C(u) > c|l_i, z)$ weight is high when l_i is high, this indicates that the downstream modules are very sensitive to the correlation between the uncertainty measure and the error, liable to yield worse performance. However, if the $P(C(u) > c|l_i, z)$ weight is low even when l_i is high, this may indicate a greater robustness of the downstream modules to the upstream uncertainty measure's incorrect ability to predict the error. Roughly speaking, in such a

case the downstream modules can be thought to be "calibrating" the uncertainty measure themselves. We illuminate these insights further with our running example.

Running Example: *Again, the type of uncertainty-aware planner that we use is the agent-avoiding planner, whose internal cost function incentivizes candidate plans that stay away from other agents by an amount proportional to the uncertainty value. We first reason about what our analysis should give us at an intuitive level, and then confirm the intuition using the formulation we have proposed.*

First, we consider using the safety cost for evaluating the system. In response to higher uncertainty values, the agent-avoiding planner prioritizes keeping a distance from the target agent and so will yield lower-cost plans in general. If the prediction has a high error, the agent-avoiding planner will have the wrong reference trajectory to keep away from. Thus, for higher error values we can expect higher-cost plans.

For the holistic cost, we can expect some of the opposite trends. As the agent-avoiding planner prioritizes staying away from the target agent, it may de-prioritize other behaviors important for driving, such as lane-keeping and reaching the goal. Thus, as the uncertainty value increases the planner will yield higher cost plans. Similar to the situation for the safety cost, as the prediction error increases we can also expect the costs of the plans to increase.

Examining Figure 4.3, we can indeed see the trends that we predict. The heights of the bars in the plots that show $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values indicate how well the planner performs under different conditions and therefore show its sensitivity or "robustness". For example, we notice that for the holistic cost evaluation, these bar heights decrease as the distance between the ego and target agent increases. This is because when the distance is high, there is less need for the agent-avoiding planner to perform as aggressive avoiding maneuvers and so its holistic driving cost does not increase as much. Thus, there is more sensitivity of the planner to scenarios with smaller ego-target agent distances. It is also interesting to note that for the holistic evaluation cost, when the prediction errors are high the bar values tend to be higher for the lower uncertainty values compared to when the prediction errors are low. There seems to be an interpretation here that the planner is also sensitive to how well the uncertainty value corresponds with the error – if higher error values do not correspond with higher uncertainty values as well, the planner does not perform as well.

For the safety evaluation cost, we see that the bar heights actually increase as the ego-agent distance increases. Often, for scenarios with high ego-agent distances, the planner does not prioritize avoidance as much and so outputs plans that are closer to the other agent and therefore less safe. We also notice that when the error is higher, the bar heights decrease as the ego-agent distance increases. This is indicative of the greater sensitivity of the planner to higher errors, but only when the ego and target agent are in close range. Otherwise, we still see the trend of higher bar values for higher errors when z is fixed.

A final remark we make here is in noting the opposing trends between using the safety cost and the holistic cost for evaluation. We consider the $P(\sigma_1, \mathbf{1}(e_1 < e_{thres})|z)$ distributions that satisfy the final specifications. For σ_1 values for which the $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ weight is high, the $P(\sigma_1, \mathbf{1}(e_1 < e_{thres})|z)$ distribution must have less probability mass in order for it to satisfy the specification. Similarly, the $P(\sigma_1, \mathbf{1}(e_1 < e_{thres})|z)$ distribution

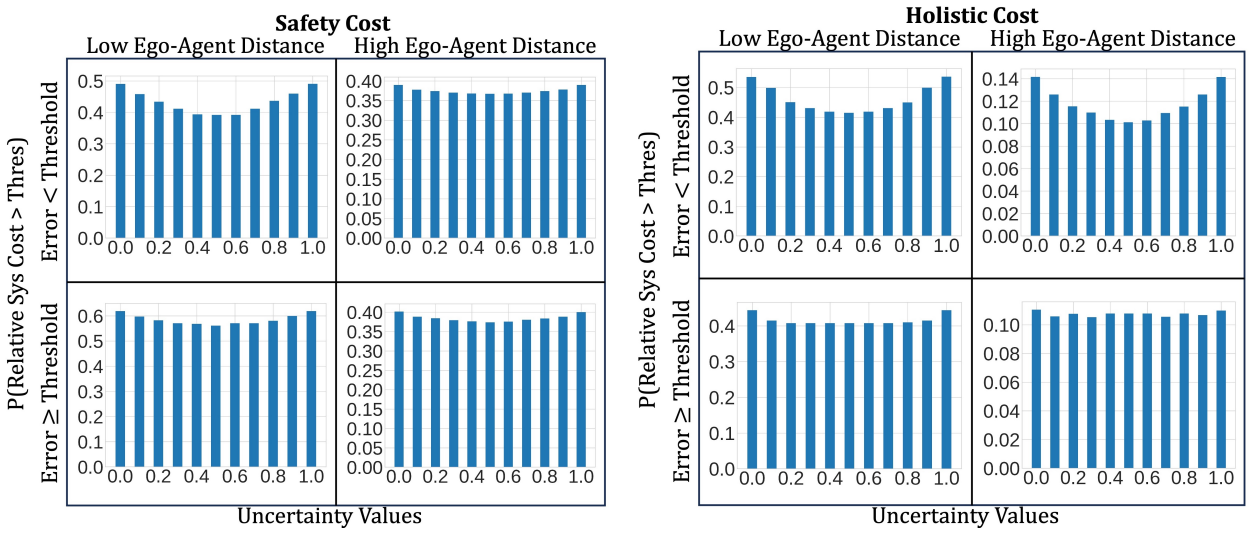


Figure 4.4: (Autonomous Driving System Case Study) For the lane-keeping planner, we show the final $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values for (left) evaluation using the safety cost and (right) evaluation using the holistic cost. For each plot, we set the values of $\mathbf{1}(e_1 < e_{thres})$ and z and show how the probability values vary by the uncertainty value. Again, the value of e_{thres} is 1.5, the value of c is 0.02, and a low ego-agent distance is defined as a distance less than 5 meters, while a high ego-agent distance is a distance greater than 5 meters. As the uncertainty value increases, the lane-keeping planner's cost does not monotonically increase for either the safety or holistic costs, since the planner is using the uncertainty value to trade off between different objectives.

can have more probability mass when the $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ weight is low. This means that for the safety cost, higher uncertainty values are permissible with higher frequency, whereas for the holistic cost, the opposite is true. We take this as an indication of the tradeoffs that can be present in decision-making under uncertainty.

Results on Case Studies

Autonomous Driving System

In this section, we show the results of our analysis for the Autonomous Driving system. Since we have already performed our analysis for the agent-avoiding planner as part of the Running Example in the previous section, here we show the analysis for the lane-keeping planner (using the internal cost function given by Eq. (2.3)). We keep the same choice of g_{enc} and z and the same technique for estimating the $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values as for the agent-avoiding planner. For both of these analyses, there are 12,452 data points in our dataset and it takes on the order of 15 minutes to obtain results on this dataset per

σ_1 grid point using a GPU. Figure 4.4 shows our results.

One of the first things to notice is the interesting U-shaped trend that we see in the $P(C(u) > c|\sigma_1, \mathbf{1}(e_1 < e_{thres}), z)$ values as σ_1 increases for most of the plots, though this trend is not as present for when the predictor error is high and the ego and target agent are far from each other. Although the shape is not perfectly symmetric, it suggests that there is an intermediate value of σ_1, roughly 0.5, at which the planner operates best for both the safety and holistic costs. If the uncertainty value is too low, the ego agent may not prioritize goal-reaching and collision avoidance enough, which results in a higher cost in terms of both safety and holistic driving. If the uncertainty value is too high, the ego agent may excessively prioritize lane-keeping, causing it to not maintain a safe enough distance from other agents and also de-prioritize goal-reaching. Thus, it seems important for the planner to be able to effectively balance all five of the internal cost objectives given in Eq. (2.1). In this way, the particular function we choose for $C^{\text{int}}_{\text{lane}}$ matters greatly; for example, when $\sigma_1 = 0$ the planner actually prioritizes lane-keeping more than when $\sigma_1 = 0.1$. When $\sigma_1 = 0.5$, for some data points the planner performs the maximally agent-avoiding maneuver. Clearly, a different method of weighting the planner internal cost terms would have demonstrated different behaviors.

Similar to the Running Example, we see trends in which the bar heights increase for the safety evaluation cost as errors increase when z is fixed. For both the safety and holistic evaluation costs, the bar heights decrease when the ego-agent distance increases. Since the bar heights for this planner are high in the first place compared to those for the agent-avoiding planner, this may be simply due to the fact that high ego-agent distances decrease cost values in general. For the holistic evaluation cost, the bar heights actually decrease as errors increase when z is fixed. This is an interesting phenomenon, which we hypothesize points to the effectiveness of a general lane-keeping strategy in the presence of higher prediction errors. Indeed, we see this effect in our robustness analysis as well, in Section 3.4.

Runway Incursion Detection System

We now pivot to discussing the Runway Incursion Detection system described in Section 2.2. We focus on σ_1, which represents the sequence of detector confidence scores corresponding to all of the detections made of a tracked object during the track's existence. Note that this means that σ_1 does not have a fixed dimension and can include fairly long sequences of numbers, for example for a track that lasts a long time.

The key elements for the setup of this case study lie in the choice of g_{enc} and z. Although this system may operate in different categories of environmental scenarios, we follow Remark 1 and Criterion (c) and find that the tracker performance can be explained solely using a particular choice of g_{enc} and without the help of any z. For g_{enc}, we follow Criterion (b) and set it to represent the *calibration error* of the detector for a particular track. Calibration error can be computed in multiple ways, for example by considering the Expected Calibration Error (ECE) [119]. Here, since we work with a bounding-box detector neural network, we compute the Detection Expected Calibration Error (D-ECE) [93] per track:

$$\sum_{j=1}^{J_{total}} \frac{|I(j)|}{N} \times |\text{avg precision}(j) - \text{avg conf}(j)| \tag{4.4}$$

where we follow [93] in partitioning the detector confidence and bounding box property spaces into bins. The total number of bins is J_{total} and j represents the index of each bin. Here, $I(j)$ is the set of detections that are in a single bin. N is the total number of detections in the track. The precision is a binary value representing whether the detection is correct or not, where correctness is determined by matching the detection with a ground-truth bounding box with sufficient IoU (Intersection over Union). The averages are taken per bin. In our analysis, we perform the binning by rounding values. Similar to the considerations for binning the l_i space as described in Section 4.5, the choices of bin sizes here are also determined by a tradeoff between the accuracy of the computation and how many samples lie in each bin. Here, we set the bin size for the confidences to 0.05 and the bin size of the box property space to 50.

For this Case Study, our specification is

$$\sum_{l_1} P(C(u) > c|l_1)P(l_1) < \alpha \tag{4.5}$$

This specifies the values that $P(l_1)$ can take, thus informing the designer on how much effort needs to be expended in calibrating the detector's uncertainty measure.

The second part of the setup for this Case Study lies in describing how we estimate the $P(C(u) > c|l_1)$ values. These values characterize the tracker module and how well it uses the uncertainty measure from the detector. While for the Autonomous Driving system we could grid the uncertainty measure space directly to generate the samples, in this case the high and variable dimensionality of σ_1 makes this difficult. Since information is lost by producing the encoding given by g_{enc}, which outputs a scalar, one cannot use samples of l_1 to produce σ_1 values and run the tracker. Therefore, our estimation technique is to run the detector and tracker module together for each tracker design for several test conditions. Here, a test condition is a particular aircraft landing scenario that lasts several minutes and that involves certain ground vehicles and results in a set of tracks being produced. Combining the data from our set of 12 test conditions, we obtain a dataset of tracks with their associated confidence scores, detections, and ground-truth bounding boxes. We compute l_1 for each track and plot these values against $C(u)$. Then, we bin the points along the l_1 axis by rounding the l_1 values. Finally, in each bin we can compute $P(C(u) > c|l_1)$ via counting. In terms of computational complexity, although each test condition is short, it can be time intensive to compute these probabilities due to overhead in running the system in ROS.

In Figure 4.5, we show both the raw data plots that we use to estimate the $P(C(u) > c|l_1)$ values as well as the $P(C(u) > c|l_1)$ values themselves. We perform these analyses for the Naive, Regression, and Histogram trackers. Similar to our analyses for the Autonomous Driving system and Running Example, we can understand these plots in terms of what they indicate about the efficacy of the trackers and in terms of the specifications they generate. We

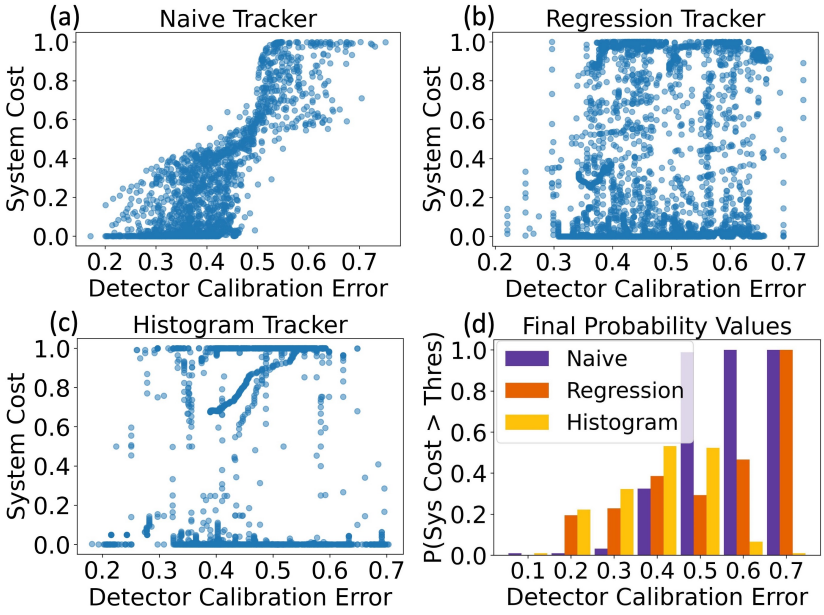


Figure 4.5: (Runway Incursion Detection System Case Study) We show the data used to estimate the $P(C(u) > c|l_1)$ values for the Naive Tracker (a), Regression Tracker (b), and Histogram Tracker (c). Each point represents a track, which corresponds to a sequence of detector confidence values. These points are binned along the Calibration Error (l_1) axis. (d) Plot showing the $P(C(u) > c|l_1)$ values for the Naive, Regression, and Histogram trackers side-by-side. The lowest bars represent a value of 0. The value of c is 0.4 and the detector calibration errors have bin sizes of 0.1.

see that as l_1 increases, the $P(C(u) > c|l_1)$ values increase as well, most sharply for the Naive tracker. Thus, we can say that the Naive tracker, while it is the simplest computationally, is the most sensitive, or least robust, to miscalibration in the detector. In comparison, the Regression and Histogram trackers produce lower $P(C(u) > c|l_1)$ weights. These trackers are taking into account a sliding window of confidence values over the track and using more sophisticated functions to compute the track's probability of existence. In this way, detector miscalibration does not affect these tracker designs as adversely.

As mentioned in our Running Example analysis, what this means for the generated specifications is that if the Naive tracker is being used downstream, less probability mass can be placed on higher values of l_1 for the $P(l_1)$ distribution. However, if the Regression or Histogram trackers are being used instead, more probability mass is permissible for higher values of l_1 for the system specification to be satisfied. In essence, using a more intelligent decision-making-under-uncertainty algorithm compensates for a more untrustworthy uncertainty measure. This analysis therefore contributes to our argument that only evaluating the calibration level of a standalone module is insufficient; by appropriately designing the entire

system, concerns about the reliability of an individual module can become less relevant.

4.6 Conclusion to Part I

In Chapters 2 and 3, we have presented both a theoretical analysis framework for understanding module-level uncertainty quantification from a system-level perspective as well as an empirical application of the analyses to two real-world systems. Our analyses are twofold: (1) We use sub-level set estimation techniques to produce a system-level robustness metric for comparing uncertainty-aware designs, and (2) We use assume-guarantee contract theory to produce a probabilistic specification on a module's uncertainty measure. Concretely, both techniques allow a designer to select between uncertainty quantification methods and decision-making algorithm designs in a principled manner and to understand why a particular design choice behaves the way it does. However, at a more conceptual level, we also position our work as providing more understanding of what an uncertainty measure truly means. At the computational level, we treat system components as black boxes to enable generality, thus shifting the analysis burden to the *data* and making our computational complexity a function of dataset sizes and simulator runtimes.

We also imagine the potential of our work to contribute to system validation processes and to producing new designs. Both of our analysis techniques involve variables that are the choice of the designer, and each setting of these variables may lead to the discovery of a different system vulnerability or behavior. The data-driven nature of our work means that the designer can determine which data points led to failure and perform further data gathering. Some of the variables and expressions introduced by our formulations could even be subject to optimization so as to produce better designs. In this way, our work can aid a system designer in producing robust-by-design architectures.

One limitation of our work lies in our analyses' assumption that the system is open-loop and time-invariant. In future work, we would like to perform a similar analysis for closed-loop systems, in which uncertainty measures may be influenced by states and observations at previous time steps.

Acknowledgments

This material is based upon work supported by the DARPA Assured Autonomy Program, the NASA ULI on Safe Aviation Autonomy, and the NSF GRFP under Grant No. 2146752. Any opinions, findings, and conclusions or recommendations expressed in this material are ours and do not necessarily reflect the views of any aforementioned organizations. Additionally, we would like to thank Brandon Schwiesow, Jose Medina, Zachary Tane, Blake Edwards, James Paunicka, and the rest of the team at Boeing for their help and contributions.

Part II

Statistical Verification

Chapter 5

Multi-Agent Reachability Calibration with Conformal Prediction

This chapter is based on work with Anish Muthali, Haotian Shen, Michael H. Lim, Rebecca Roelofs, Aleksandra Faust, and Claire J. Tomlin in [118].

5.1 Introduction

In safety-critical situations in which an autonomous agent interacts with humans, it is often necessary to predict human behavior in order for the agent to appropriately react. For example, in self-driving tasks, the autonomous car is responsible for assuring the safety of itself and the other vehicles it encounters. State-of-the-art systems try to achieve this through behavior prediction and motion forecasting models, oftentimes black-box neural networks that do not provide interpretation of their inner workings [68, 132, 152]. However, these models lack rigorous safety assurances, especially in the presence of data distribution shifts. While lower dimensional models can provide safety assurances, these methods rely on parametric assumptions on, for example, a human's rationality [61]. These assumptions can be inadequate for complex, multi-modal, and noisy decision-making scenarios. In this work, we provide safety assurances for state-of-the-art black-box trajectory forecasting methods by quantifying these models' uncertainty.

However, uncertainty quantification methods alone are insufficient for providing safety assurances. This is because they do not provide guarantees on model behavior and may be unreliable or uninterpretable [31, 86, 166]. Additionally, using prediction model uncertainty for assuredly safe decision-making in downstream planning and control is a challenging problem.

In our work, we first utilize conformal prediction to *calibrate* measures of uncertainty [154]. Conformal prediction is a statistical tool that uses a heuristic notion of risk to non-parametrically estimate quantiles of risk given a sequence of past observations [9]. Existing methods that leverage conformal prediction in the context of trajectory forecasting do not ex-

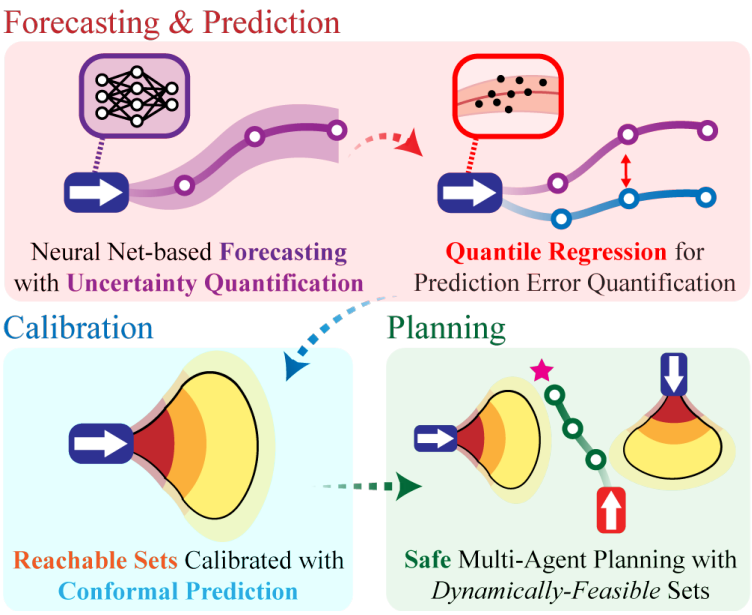


Figure 5.1: Our method generates dynamically-feasible, probabilistic confidence sets that are derived from a conformal-calibrated quantile regression model.

plicitly use interpretable metrics of prediction uncertainty [33, 49, 107, 110, 151]. We propose a method that provides rigorous confidence intervals on model error, a form of probabilistic assurance, given any interpretable heuristics on a trajectory forecasting model's prediction uncertainty. Our approach also allows us to examine the efficacy of various uncertainty quantification heuristics when attempting to predict model error. In addition, we extend our analysis to multi-agent environments to closely reflect real-world assurance cases. We analyze the problem of a single autonomous agent, commonly described as an "ego agent", interacting with other, uncontrolled agents.

By producing estimates of model error, we are able to couple statistical assurances with dynamical assurances to allow for safe downstream navigation. In particular, we turn to Hamilton-Jacobi (HJ) reachability analysis [15], which provides guarantees on dynamical systems by means of reachable sets and associated controllers. In HJ reachability, a Hamilton-Jacobi partial differential equation is solved to obtain an optimal value function and controller. The sub-zero level sets of this value function are the reachable sets (possible states of an agent at a given time) and tubes (possible states of an agent *up to* and including a given time).

Our method can be outlined as follows: given a trajectory forecasting model with an associated uncertainty heuristic, we design a quantile regression model that correlates uncertainty with prediction error, creating an approximate confidence interval on the model's prediction. We then apply conformal prediction to calibrate the confidence intervals and

provide guarantees on miscoverage rate. We map the calibrated intervals in control action space to sets in state space through reachability analysis, and we demonstrate the utility of these confidence sets in planning tasks. Thus, this chapter contributes:

1. A novel way to interpret trajectory forecasting models' prediction uncertainty and obtain approximate confidence intervals (Section 5.3);
2. A technique to calibrate the aforementioned intervals using conformal prediction (Section 5.3);
3. Dynamically-feasible, probabilistic reachable sets using calibrated intervals (Section 5.4);
4. A planning framework that leverages assurances developed in the previous steps (Section 5.4).

Section 5.2 discusses related works in conformal prediction and assurances in trajectory forecasting models, Section 5.3 and Section 5.4 describe the contributions outlined above, and Section 5.5 showcases the safety and performance of our methods compared to baseline methods.

5.2 Related Works

Conformal Prediction

Conformal prediction [9, 154] is a class of uncertainty quantification methods for constructing prediction sets that satisfy a significance level (false negative rate) requirement. Traditionally, conformal prediction creates empirical histograms of measures of risk, called nonconformity scores, and uses these to estimate prediction intervals. Classical techniques include split conformal prediction, which creates empirical histograms from hold-out sets, and full conformal prediction, which creates empirical histograms using all available data [154]. Inductive conformal prediction, a variant of split conformal prediction, uses a non-conformity score that measures distance between train and test data [22]. These methods require that the data are identically distributed and exchangeable (any permutation of data points are identically distributed). Methods such as [148] relax the requirement for identically distributed data, and [16, 162] relax the exchangeability requirement. Adaptive Conformal Inference [65] and Rolling Risk Control (RollingRC) [56] have been proposed to relax all assumptions by further calibrating the significance level to match a desired error rate. We adapt RollingRC to provide safety assurances in any multi-agent scenario.

Probabilistic Reachability Frameworks

In this work, we introduce a method to generate probabilistic reachable sets to account for agents' dynamics. Previous work in this space typically involves randomly generating

inputs and observing corresponding outputs of a dynamics model, with some associated guarantees in the sampling process [44]. Other methods, much like ours, leverage neural network uncertainty [120]. Specifically, the method of Nakamura and Bansal [120] uses Gaussian mixture models (GMMs) output by a trajectory forecasting model to generate parametric confidence intervals on control actions, which are then used as control bounds in reachability calculations. In this work, we attempt to relax assumptions that the control actions follow any parametric distribution by applying non-parametric inference techniques.

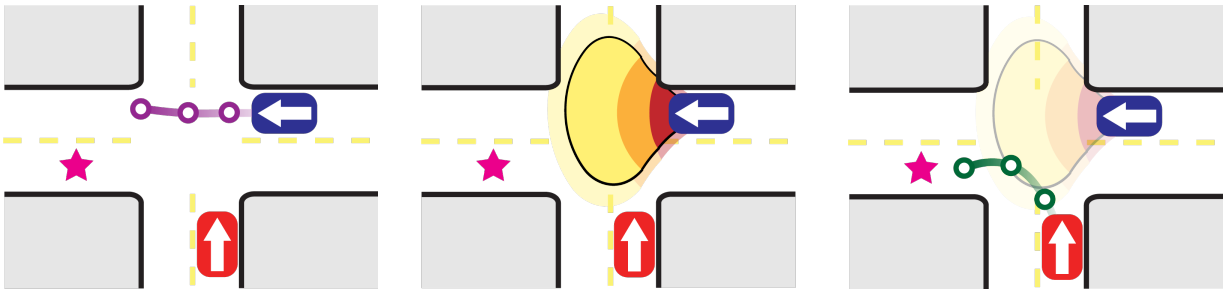
Safety Assurances in Trajectory Prediction

Various methods for incorporating uncertainty quantification have been examined for the purposes of providing safety assurances in trajectory prediction problems. Some methods provide probabilistic assurances by inferring parameters of a distribution on an agent's control actions [11, 61]. Methods such as [49] and [110] estimate confidence intervals with conformal prediction, implicitly leveraging prediction uncertainty through the non-conformity measure. Specifically, the method of Luo et al. [110] uses split conformal prediction to create a warning system, alerting drivers of "dangerous" situations. These warnings can be transformed into confidence sets, as shown in [49], which additionally eliminates exchangeability assumptions and incorporates trajectory optimization, much like our approach. Other methods emphasize the design process of the trajectory prediction neural networks, for instance by opting to use ReLU networks [33] or by opting to incorporate conformal prediction in the neural network's loss function [151]. In contrast to our approach, none of these methods consider the dynamic feasibility of confidence sets, and some methods that investigate conformal prediction, such as [33] and [110], assume exchangeability. Our method relaxes these assumptions while providing interpretability in the uncertainty quantification process and dynamic feasibility in confidence sets.

5.3 Assurances from Uncertainty

Our approach can be summarized in four primary steps: trajectory forecasting with uncertainty quantification, leveraging uncertainty to obtain approximate prediction intervals, calibrating approximate prediction intervals, and obtaining dynamically feasible prediction sets. We summarize our algorithm and we apply our approach to ego agent planning tasks in Section 5.4.

Running Example: *To motivate and illustrate our method, we introduce a simple running example with two vehicles at an intersection, one of them designated as the "ego" vehicle. In Fig. 5.2, the autonomous ego vehicle, shown in red, aims to safely navigate to the pink-colored star while avoiding the blue vehicle.*



(a) Output of Trajectron++ in a simple scene with two vehicles.

(b) Approximate (opaque) and calibrated (translucent) reachable sets.

(c) Probabilistically-safe plan generated by the reachability-based planner.

Figure 5.2: Visualization of the running example. The autonomous ego vehicle is shown in red, and the human driver is shown in blue. The ego vehicle aims to navigate to the pink star while avoiding a collision with the human-driven vehicle. Confidence sets for the next three prediction steps are shown. In (b), the redder regions represent confidence sets for earlier prediction timesteps, and the translucent regions represent conformal prediction's calibration effect.

Trajectory Forecasting Model

We start by assuming access to a known dynamics model for each agent and a trajectory forecasting model capable of predicting an agent's control input. This trajectory forecasting model may maintain the capability to predict control actions for multiple agents at once, while considering interactions between agents. We denote the model as $f_T(\cdot) : \mathcal{X} \to \mathcal{U}$, where $\mathcal{X}$ is some arbitrary input space and $\mathcal{U}$ is a space over control actions. The network predicts $\mathbf{u}_{t:t+h} \in \mathcal{U}$, which is a collection of control action vectors indexed by timesteps t through $t+h$, for each of N total agents. Here, h is a fixed prediction horizon. We also assume the existence of an uncertainty measure on the network's outputs, denoted as $\sigma_T(\cdot) : \mathcal{X} \times \mathcal{U} \to \mathbb{R}^d$, with d as the dimension of uncertainty representation. For example, a variance prediction or a variance estimate from inference-time dropout [86] is a valid uncertainty measure. Additionally, some neural network architectures, such as Trajectron++ [132], provide alternative uncertainty measures. This network architecture predicts a GMM over possible control actions, leading to understandings of prediction uncertainty such as the variance of the GMM's modes.

Running Example: *Given some sequence of the other vehicle's position history, Trajectron++, our trajectory forecasting model of choice, predicts a GMM in action space, and then integrates the actions to obtain states. We assume that vehicles follow the extended Dubins' car dynamics model. The state of this system is* $\mathbf{x} = \begin{bmatrix} x & y & v & \theta \end{bmatrix}^\top$*, and the dynamics are given by* $\dot{\mathbf{x}} = \begin{bmatrix} v\cos(\theta) & v\sin(\theta) & u_1 & u_2 \end{bmatrix}^\top$*. The variance of the highest-probability Gaussian component, among other features of the GMM, are incorporated into the design of the uncertainty measure.*

Estimating Model Error from Uncertainty

To obtain confidence intervals on a black-box model's outputs, we estimate the neural network's confidence in an online manner, correlating its prediction uncertainty with prediction error. Quantile regression models enable us to map heuristic notions of uncertainty to an *approximate* confidence interval [92] along each action dimension. We choose a linear model since its simple parametrization allows for fast online updates and interpretability in how it perceives uncertainty. We demonstrate an example of interpretability in Section 5.5. Intuitively, our quantile regression models are approximately "calibrating" the network's uncertainty to obtain an estimate of its error.

As we observe new datapoints online, we collect $\mathbf{u}_{t-h:t}$, the last h ground truth control actions prior to timestep t. In practice, we estimate control actions by observing the state history of an agent, and then numerically computing derivatives to estimate actions from an assumed dynamics model. We contrast $\mathbf{u}_{t-h:t}$ with the network's previous prediction h timesteps ago, i.e., $\widehat{\mathbf{u}}_{t-h:t}$, and define $\mathbf{e}_{t-h:t} \coloneqq \mathbf{u}_{t-h:t} - \widehat{\mathbf{u}}_{t-h:t}$ as the prediction error.

Now, suppose we require a $1-\alpha$ approximate confidence interval on the ground truth control action. We can construct two quantile regression models $\widehat{q}_{\frac{\alpha}{2}} : \mathbb{R}^d \to \mathcal{U}$ and $\widehat{q}_{1-\frac{\alpha}{2}} : \mathbb{R}^d \to \mathcal{U}$ where $\widehat{q}_\varepsilon$ estimates the ε-quantile on the network's prediction error from timesteps t to $t+h$ for each agent, denoted $\widehat{\mathbf{e}}_\varepsilon$. For notational convenience, let us denote $\mathbb{P}_t(A)$ as the probability of event A conditioned on information until time t. We obtain an approximate $1-\alpha$ confidence interval as follows:

$$\begin{aligned}
&\mathbb{P}_t(\widehat{\mathbf{e}}_{\frac{\alpha}{2}} \leq \mathbf{e}_{t:t+h} \leq \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}}) && (5.1)\\
&= \mathbb{P}_t(\widehat{\mathbf{e}}_{\frac{\alpha}{2}} \leq \mathbf{u}_{t:t+h} - \widehat{\mathbf{u}}_{t:t+h} \leq \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}}) && (5.2)\\
&= \mathbb{P}_t(\widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{\frac{\alpha}{2}} \leq \mathbf{u}_{t:t+h} \leq \widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}}) && (5.3)\\
&\approx 1-\alpha. && (5.4)
\end{aligned}$$

Thus, our approximate $1-\alpha$ confidence interval on $\mathbf{u}_{t:t+h}$ is $\widehat{\mathcal{I}}_{t:t+h} = [\widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{\frac{\alpha}{2}}, \widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}}]$.

Traditionally, quantile regression models are trained using computationally expensive linear programs [92], so instead, we opt for a faster, online gradient descent approach. We define our loss function for the quantile regression model $\widehat{q}_\varepsilon$ to be $\mathcal{L}(y, \widehat{y}) = (y - \widehat{y})\varepsilon \mathbf{1}\{y \geq \widehat{y}\} + (\widehat{y} - y)(1-\varepsilon)\mathbf{1}\{y < \widehat{y}\}$ (the "pinball loss") [143]. We set y to be the true model error, $\mathbf{e}$, and $\widehat{y} = \boldsymbol{\beta}^\top \boldsymbol{\sigma}$, where $\boldsymbol{\beta}$ represents the weights of the regression model, and $\boldsymbol{\sigma}$ is the uncertainty measure from the trajectory forecasting model. We update the weights according to $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} - \zeta \nabla_{\boldsymbol{\beta}} \mathcal{L}(\mathbf{e}_{t-h:t}, \boldsymbol{\beta})$, with learning rate ζ.

Calibrating Approximate Confidence Intervals

Given that the confidence intervals we obtained in the previous section are merely approximate, we aim to calibrate these intervals. To this end, we apply the RollingRC algorithm [56], which perfectly adapts to the online requirements of our method. We are motivated to use the RollingRC algorithm compared to other conformal prediction methods due to a desire

to remove the data exchangeability assumption, since we allow for sequentially-dependent data and potential distribution shifts. In addition, we would like to train and calibrate the quantile regression models in a sample-efficient manner. RollingRC guarantees that the error rate deviates from α as $\mathcal{O}(1/T)$, where T is the total number of datapoints provided to the algorithm.

Following the notation from the RollingRC algorithm, we define $\theta_t \in \mathbb{R}$ as our conformal parameter, and $\varphi(\cdot) : \mathbb{R} \to \mathcal{U}$ as the algorithm's "stretching function". Now, we claim that

$$\begin{aligned} &\mathbb{P}_t(\widehat{\mathbf{e}}_{\frac{\alpha}{2}} - \varphi(\boldsymbol{\theta}) \leq \mathbf{e}_{t:t+h} \leq \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}} + \varphi(\boldsymbol{\theta})) \\ &\quad \geq 1 - \alpha - \mathcal{O}(1/t). \end{aligned} \tag{5.5}$$

Following similar steps as before, we obtain our newly calibrated confidence interval on $\mathbf{u}_{t:t+h}$ as $\mathcal{I}_{t:t+h} = [\widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{\frac{\alpha}{2}} - \varphi(\boldsymbol{\theta}), \widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}} + \varphi(\boldsymbol{\theta})]$.

5.4 Probabilistic Reachability and Planning

Probabilistic Reachability among Multiple Agents

In the previous sections, we have designed a method to provide confidence intervals on agents' control actions. However, for some downstream tasks, such as safe planning, confidence sets in spatial dimensions are more desirable. Hence, we use HJ reachability to obtain spatial sets, in the form of forward reachable tubes, on each agent's location given its dynamics and the probabilistic bound on control [136]. This procedure asserts that an agent's location will be contained in the produced reachable tube with probability $1 - \alpha$.

Suppose we wish to upper bound the probability that the ego vehicle collides with any agent. Let $\mathbf{x}_t^{(i)}$ be the location of non-ego agent i at timestep t and $\mathcal{S}[t]^{(i)}$ be the corresponding agent's forward reachable tube, as computed by our algorithm. We define miscoverage rate as the proportion of instances in which the ground truth position of any agent i at time t is outside $\mathcal{S}[t]^{(i)}$. We aim to obtain an upper bound on miscoverage rate, such that the ego agent can navigate in regions outside of $\mathcal{S}[t]^{(i)}$ for all $i \in \{1, \ldots, N\}$ and guarantee that the probability of collision is at most γ, a pre-specified parameter. Consequently, we set the confidence interval significance level α according to our desired total miscoverage rate γ and number of agents N.

Theorem 1 (Significance Level Correction)**.** *Suppose that we wish to have a total miscoverage rate of γ, where total miscoverage rate is an upper bound on the probability that any human agent is miscovered:*

$$\mathbb{P}_t\Big(\bigcup_{i=1}^{N} \big\{ \mathbf{x}_t^{(i)} \notin \mathcal{S}[t]^{(i)} \big) \leq \gamma. \tag{5.6}$$

We claim that the following α achieves an (asymptotic) total miscoverage rate of γ for N human agents:

$$\alpha = 1 - (1 - \gamma)^{\frac{1}{N}}. \tag{5.7}$$

The proof of Theorem 1 is available in the Appendix at the end of this chapter, and uses the fact that the N agents act independently conditioned on past information. Since α must be a fixed quantity in our algorithm, we must also fix N. Hence, we fix our algorithm to only consider the N agents closest to the ego vehicle.

Running Example: *Suppose we want a 95% probability safety assurance. Since there is only one other vehicle, we get* $\alpha = 0.05$ *from Theorem 1. Given the previous predictions of the blue agent's trajectory, we generate uncalibrated, time-indexed intervals on ranges of possible control actions, denoted* $\widehat{\mathcal{I}}_t, \widehat{\mathcal{I}}_{t+\Delta t}, \widehat{\mathcal{I}}_{t+2\Delta t}$*. We calibrate these using conformal prediction to obtain* $\mathcal{I}_t, \mathcal{I}_{t+\Delta t}, \mathcal{I}_{t+2\Delta t}$*. As we explain in the next subsection, HJ reachability allows us to take any sequence of intervals on control actions and generate a time-indexed set of states. In Fig. 5.2.b, we distinguish the effects of quantile regression and RollingRC's calibration.*

Full Algorithm

In Algorithm 1, we demonstrate the final algorithm to generate probabilistic reachable sets. The HJReachability function generates reachable sets given a probabilistic range of control actions, $\mathcal{I}_{t:t+\Delta t}$. Since the range can differ over time (e.g., $\mathcal{I}_t \neq \mathcal{I}_{t+\Delta t}$ necessarily), we iteratively compute time-indexed forward reachable tubes by computing the forward reachable tube over $[t, t+\Delta t]$ and using the reachable *set* at $t+\Delta t$ as the initial condition to compute the reachable tube over $[t+\Delta t, t+2\Delta t]$. We also utilize a GradientDescent function that updates the weights of the quantile regression models as described in Section 5.3. In Algorithm 1, ξ is the "learning rate" associated with the RollingRC algorithm.

Safe Planning Framework

Given the time-indexed sets $\mathcal{S}[t] \subseteq \mathcal{S}[t+\Delta t] \subseteq \cdots \subseteq \mathcal{S}[t+h]$, we desire that the autonomous agent's location at time t' is outside $\mathcal{S}[t+k\Delta t]$, where $t+(k-1)\Delta t \leq t' \leq t+k\Delta t$. We can plan by treating each agent's time-indexed forward reachable tube as a dynamic obstacle that grows with time. The obstacle-aware planning requirement motivates the application of a forward reach-avoid tube for the ego agent [15, 58]. We use this to derive an optimal control trajectory by selecting the Hamiltonian-maximizing control trajectory to a desired final state within the forward reach-avoid tube. In practice, this trajectory can involve bang-bang control, so one can track it using a tracker with a provable tracking error bound, such as a constrained iterative linear quadratic regulator. The planner's output is visualized in the Appendix at the end of this chapter.

Running Example: *From the previous section, we obtained* $\mathcal{S}[t], \mathcal{S}[t+\Delta t], \mathcal{S}[t+2\Delta t]$ *as a probabilistic occupancy region on the location of the other vehicle. Now, we can use the time-varying avoidance regions to plan a safe path to the goal in Fig. 5.2.c. Notice that the planner allows the ego agent to traverse in the yellow-colored region: it is aware that the ego vehicle would not violate the safety assurance as it can leave the yellow region by the time the other agent would enter it.*

Algorithm 1: Conformal Reachability Calibration

procedure GENERATESETS($\boldsymbol{\theta}$, $\widehat{\mathbf{u}}_{t:t+h}$, $\boldsymbol{\sigma}$)
  $\widehat{\mathbf{e}}_{\frac{\alpha}{2}} \leftarrow \widehat{q}_{\frac{\alpha}{2}}\{\boldsymbol{\sigma}\}$ *// Obtain lower $\frac{\alpha}{2}$ quantile*
  $\widehat{\mathbf{e}}_{1-\frac{\alpha}{2}} \leftarrow \widehat{q}_{1-\frac{\alpha}{2}}\{\boldsymbol{\sigma}\}$ *// Obtain upper $\frac{\alpha}{2}$ quantile*
  $\mathcal{I}_{t:t+h} \leftarrow \left[\widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{\frac{\alpha}{2}} - \varphi(\boldsymbol{\theta}), \widehat{\mathbf{u}}_{t:t+h} + \widehat{\mathbf{e}}_{1-\frac{\alpha}{2}} + \varphi(\boldsymbol{\theta})\right]$
  $\mathcal{S} \leftarrow []$
  for $t' \in \{t, t+\Delta t, \ldots, t+h\}$ **do**
    $\mathcal{S}[t'] \leftarrow$ HJREACHABILITY($\mathcal{I}_{t'}$)
  end for
  return $\mathcal{S}, \mathcal{I}_{t:t+h}$
end procedure
procedure UPDATE($\boldsymbol{\theta}$, $\mathbf{u}_{t-h:t}$, $\mathcal{I}_{t-h:t}$)
  for $t' \in \{t, t+\Delta t, \ldots, t+h\}$ **do**
    $\boldsymbol{\theta}\{t'\} \leftarrow \boldsymbol{\theta}\{t'\} + \xi(\mathbf{1}\{\mathbf{u}_{t'-h} \notin \mathcal{I}_{t'-h}\} - \alpha)$
  end for
  GRADIENTDESCENT($\widehat{q}_{\frac{\alpha}{2}}$, $\mathbf{u}_{t-h:t}$)
  GRADIENTDESCENT($\widehat{q}_{1-\frac{\alpha}{2}}$, $\mathbf{u}_{t-h:t}$)
  return $\widehat{q}_{\frac{\alpha}{2}}$, $\widehat{q}_{1-\frac{\alpha}{2}}$, $\boldsymbol{\theta}$
end procedure
procedure MAIN(γ, N)
  $\alpha \leftarrow 1 - \{1-\gamma\}^{\frac{1}{N}}$
  $\boldsymbol{\theta}\{t, t+\Delta t, \ldots, t+h\} \leftarrow 0$
  $\widehat{q}_{\frac{\alpha}{2}}, \widehat{q}_{1-\frac{\alpha}{2}} \leftarrow$ INITIALIZERANDOMWEIGHTS()
  $\mathbb{I} \leftarrow \{\}$
  $t \leftarrow 0$
  while true **do**
    $\widehat{\mathbf{u}}_{t:t+h}, \boldsymbol{\sigma} \leftarrow f_T\{\cdot\}, \sigma_T\{\cdot\}$ *// Get trajectory predictions and uncertainty from model*
    $\mathcal{S}, \mathcal{I}_{t:t+h} \leftarrow$ GENERATESETS($\boldsymbol{\theta}$, $\widehat{\mathbf{u}}_{t:t+h}$, $\boldsymbol{\sigma}$)
    $\mathbb{I} \leftarrow \mathbb{I} \cup \mathcal{I}_{t:t+h}$
    if $t \geq h$ **then**
      $\mathbf{u}_{t-h:t} \leftarrow$ OBSERVEHISTORY()
      $\mathcal{I}_{t-h:t} \leftarrow \mathbb{I}[t-h:t]$
      $\widehat{q}_{\frac{\alpha}{2}}$, $\widehat{q}_{1-\frac{\alpha}{2}}$, $\boldsymbol{\theta} \leftarrow$ UPDATE($\boldsymbol{\theta}$, $\mathbf{u}_{t-h:t}$, $\mathcal{I}_{t-h:t}$)
    end if
    $t \leftarrow t + \Delta t$
  end while
end procedure

5.5 Results

We compare the empirical safety and efficiency of our contribution to two baselines, Online Update of Safety Assurances Using Confidence-Based Predictions by Nakamura and Bansal [120] and Sample-Efficient Safety Assurances using Conformal Prediction by Luo et al. [110]. For both baselines, we perform the significance level correction described in Section 5.4.

For all benchmarking purposes, we use Trajectron++ trained on the relevant datasets. We follow the same architecture and hyperparameters as [132] by using 4 seconds (8 steps)

	Coverage Rates for Prediction Step					
Methods	1st (.5s)	2nd (1s)	3rd (1.5s)	4th (2s)	5th (2.5s)	6th (3s)
nuScenes Dataset						
Nakamura and Bansal	0.926 ±0.012	0.854 ±0.017	0.816 ±0.023	0.842 ±0.023	0.868 ±0.022	0.902 ±0.020
Luo et al.	1.000 ±0.000	1.000 ±0.000	1.000 ±0.000	0.998 ±0.002	0.989 ±0.006	0.968 ±0.012
Our Method	0.964 ±0.008	0.962 ± 0.011	0.968 ±0.010	0.975 ±0.008	0.981 ±0.007	0.985 ±0.007
Waymo Dataset						
Nakamura and Bansal	0.981 ±0.007	0.954 ±0.012	0.938 ±0.014	0.937 ±0.015	0.952 ±0.014	0.955 ±0.015
Luo et al.	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	0.998 ±0.002	0.985 ±0.009	0.955 ±0.015
Our Method	0.997 ±0.002	0.986 ±0.006	0.980 ±0.008	0.967 ±0.011	0.965 ±0.011	0.965 ±0.013
	Set Sizes for Prediction Step					
nuScenes Dataset						
Nakamura and Bansal	57 ±2	320 ±12	886 ±35	2060 ±85	3259 ±128	4285 ±160
Luo et al.	425 ±11	523 ±16	683 ±34	1097 ±109	1426 ±140	1814 ±186
Our Method	39 ±3	157 ± 13	462 ±39	1078 ±88	2150 ±170	3713 ±268
Waymo Dataset						
Nakamura and Bansal	64 ±7	311 ±32	951 ±96	2117 ±195	3814 ±328	5892 ±475
Luo et al.	448 ±8	736 ±43	1110 ±94	1568 ±169	2126 ±237	2687 ±301
Our Method	61 ±6	246 ±27	655 ±71	1361 ±143	2422 ±240	3885 ±365

Table 5.1: Coverage Rates and Set Sizes for $1 - \gamma = 0.95$.

of history to predict 3 seconds (6 steps) into the future. This is consistent with the other baselines' approaches. Set sizes are shown in square meters. We use a pre-specified total miscoverage rate of $\gamma = 0.05$, and we generate predictions for the closest $N = 3$ agents, which strikes a balance between the speed of our HJ reachability calculations and the practical safety of the system.

nuScenes Dataset Results

We compare the coverage rate and efficiency of our method against the two baselines on nuScenes self-driving data [24]. We calculate average coverage rate and average set sizes individually for each forward prediction step $t, t + \Delta t, \ldots, t + h$ on 100 randomly sampled scenes. For each scene, we use the first 13 seconds to calibrate each method and make predictions on the last 5.5 seconds. Table 5.1 shows step coverage and set sizes at all prediction steps. Note that an ideal algorithm maintains a coverage rate over $1 - \gamma$ while providing the smallest prediction sets.

Waymo Open Motion Dataset Results

To demonstrate the planning safety and efficiency of each method, we also perform experiments on the Waymo Open Motion Dataset [53], coupled with the Nocturne simulator [153]. This allows us to apply control actions to the ego vehicle while all other agents replay their respective sequences of control actions from the dataset. We use the same planning method discussed in Section 5.4 for all three methods, since neither of the baselines have associated planners. For each scene, we calibrate using the first 7 seconds and use model-predictive control to plan for the last 3 seconds, where the goal is the final position of the ego vehicle in the ground truth data. We measure three quantities: (1) progress to goal, defined as the ratio of the distance from the final state of the ego vehicle to the goal compared to the distance from the start to the goal, subtracted from 1; (2) collision rate; (3) conservatism of each method compared to the ego vehicle's ground truth trajectory, defined as the ratio of minimum distance between the ego vehicle to other agents at all times as a result of the planner, compared to that of the ground truth. The formulas and computations of these metrics are described in detail in the Appendix at the end of this chapter. In the Appendix, we additionally demonstrate the impact of the aforementioned theoretical guarantees by providing safety and efficiency metrics in the absence of conformal prediction. We performed the benchmarks on 200 randomly sampled scenes. Table 5.1 depicts coverage rates and set sizes for all prediction timesteps. Table 5.2 depicts average collision rate, average progress to goal, and conservatism.

Discussion of Results

For the nuScenes dataset, we notice that our method achieves more efficient set sizes for initial prediction steps, while Luo et al. achieves more efficient set sizes for later prediction

Table 5.2: Waymo Planning Benchmarks

Method	Progress to Goal	Collision Rate	Conservatism
Nakamura and Bansal	0.494 ±0.029	**0.0**	**1.504** ±0.068
Luo et al.	0.305 ±0.028	0.005	1.626 ±0.072
Our Method	**0.544** ±0.028	**0.0**	1.507 ±0.068

timesteps. Nevertheless, neither of these two methods violates the miscoverage requirement of $\gamma = 0.05$. The method of Nakamura and Bansal violates the miscoverage rate, however, supporting the introduction of uncertainty calibration into the algorithm. Hence, calibrating neural network uncertainty is important, not only to provide the desired coverage rate but also to generate efficient prediction sets.

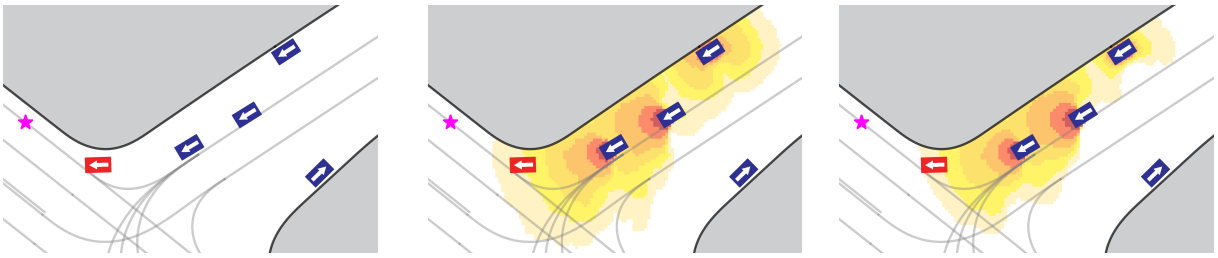
For the Waymo dataset, we notice a very similar phenomenon with set sizes and coverage rates. In the planning benchmarks, our method has the best progress to goal, likely due to the initial-timestep sets being smaller. This is also reflected in the conservatism scores, with reachability-based methods performing the best. The method of Luo et al. also encountered one collision scenario in which the produced set was very large and forced the planner to take a sharp avoid action. Thus, we note the importance of initial-timestep sets being small to allow the reachability-based methods to perform better in the planning benchmarks. This allows the ego vehicle to make some progress, whereas a large initial-timestep set would inhibit any progress regardless of the relative size of later timesteps' sets.

Case Studies

Understanding Uncertainty Measures

In this case study, we demonstrate the usefulness of our interpretable quantile regression model when understanding the efficacy of uncertainty metrics. Consider the scene in Fig. 5.3.a. We choose the uncertainty measure based on properties of the GMM, including the distance between peaks and the (co)variance of the highest-weighted mode. The learned regression model indicates a positive correlation between prediction error and variance of the most-likely GMM mode. In Fig. 5.3.b and Fig. 5.3.c, we can visually discern the benefit of including these features.

Overall, this case study shows the importance of understanding the usefulness of different components of the uncertainty measure. A more useful uncertainty metric can provide more efficient sets, since a more accurate quantile regression model would require less calibration (less "stretching" from conformal prediction). Conformal prediction cannot derive confidence intervals conditional on some input, so quantile regression's accuracy is crucial for providing efficient sets.



(a) Sample scenario in which we observe the reachable sets of the three agents closest to the ego vehicle.

(b) Calibrated confidence sets generated by quantile regression **without** covariance features.

(c) Calibrated confidence sets generated by quantile regression **with** covariance features.

Figure 5.3: Case Study of Uncertainty Metrics. We demonstrate a simple example in which the choice of uncertainty measure affects the size of sets, with coverage rate held constant.

Safety in Aerospace Applications

In this case study, we apply our algorithm to satisfy a real-world safety assurance requirement by demonstrating our algorithm on Boeing vehicles. We consider the case of an aircraft attempting to land on a runway while accounting for potential runway incursions from ground vehicles. We use our algorithm to provide assurances on the motion of a ground vehicle on the runway. Given a fixed landing plan for the plane, we adapt the sets from our algorithm to design a warning system similar to Luo et al.'s original algorithm. If a prediction set intersects the runway, a warning is issued. A visualization of this application is shown in Fig. 5.4. The ground vehicle's state history is shown in red, and its uncalibrated prediction set is shown in purple. The "stretching" effect from conformal prediction is shown in orange.

5.6 Discussion and Future Work

This chapter introduced a non-parametric approach to using interpretable uncertainty measures from black-box models for generating calibrated prediction intervals. We demonstrated an efficient reachability-based approach to generating prediction sets, and we showed the goal-oriented efficiency and safety of our algorithm in planning tasks for an ego agent through simulations and real-world experiments.

For future investigations, we would be interested in seeing the effects of longer planning horizons. Although many state of the art models cannot provide reliable predictions for agent behavior 1 minute into the future, for example, we would like to see the efficiency of our method compared to the other methods discussed. We would also like to generalize our method to arbitrary measures of risk, instead of only coverage rate. For example, one might want greater confidence in the behavior of nearby or fast-moving agents, than for agents that are far away or stationary. Thus, in certain practical scenarios, a heuristic measure of



Figure 5.4: Our algorithm is applied to assure safety in potential runway incursion scenarios. Once the ground vehicle is determined to have crossed a designated safety threshold, the aircraft is cleared to land.

risk may be more appropriate than miscoverage rate. Along these lines, we would also like to explore adapting the confidence interval significance α for different agents depending on their properties with respect to maintaining safety. Finally, we are interested in decreasing conservatism across the pipeline to help ensure that the planning framework can always find a feasible solution to the problem.

Acknowledgements

This material is based upon work supported by the DARPA Assured Autonomy Program, the SRC CONIX program, Google-BAIR Commons, the NASA ULI program on Safe Aviation Autonomy, and the National Science Foundation Graduate Research Fellowship Program under Grant Nos. DGE 1752814 and DGE 2146752. Any opinions, findings, and conclusions or recommendations expressed in this material are ours and do not necessarily reflect the views of any aforementioned organizations. We also thank James Paunicka, Blake Edwards, Dragos Margineantu, Douglas Stuart, and the entire team at Boeing for all their help and contributions in demonstrating our method on Boeing's runway incursion dataset. Finally, we thank Marius Wiggert for helping with the implementations in the HJ reachability toolbox.

5.7 Appendix

Proof of Theorem 1

The theorem is restated for convenience below:

Theorem 1 (Significance Level Correction)**.** *Suppose that we wish to have a total miscoverage rate of γ, where total miscoverage rate is an upper bound on the probability that any human agent is miscovered:*

$$\mathbb{P}_t\Big(\bigcup_{i=1}^{N}\big\{\mathbf{x}_t^{(i)} \notin \mathcal{S}[t]^{(i)} \quad\big)\Big) \leq \gamma. \tag{5.8}$$

We claim that the following α achieves an (asymptotic) total miscoverage rate of γ for N human agents:

$$\alpha = 1 - (1-\gamma)^{\frac{1}{N}}. \tag{5.9}$$

Proof. To obtain this, we assume that each agent reacts to other agents entirely based on past observations. Mathematically, we define a filtration over the probability space of all agents' time-dependent actions $\mathcal{F}_0 \subset \mathcal{F}_{\Delta t} \subset \ldots \subset \mathcal{F}_{t-\Delta t} \subset \mathcal{F}_t$. We assume that any probability event corresponding to agent i's future behavior, denoted $A_{t'}^{(i)}$, is conditionally independent of any probability event corresponding to agent j's future behavior, denoted $A_{t'}^{(j)}$. Implicitly, we write $t' > t$. Specifically,

$$\mathbb{P}(A_{t'}^{(i)} A_{t'}^{(j)} | \mathcal{F}_t) = \mathbb{P}(A_{t'}^{(i)} | \mathcal{F}_t)\mathbb{P}(A_{t'}^{(j)} | \mathcal{F}_t). \tag{5.10}$$

Intuitively, no agent makes decisions on unseen observations. Hence, we may rewrite Eq. 5.8 as follows using the conditional independence and the corresponding lower bound on coverage rate for each agent:

$$\mathbb{P}\Big(\bigcup_{i=1}^{N}\big\{\mathbf{x}_t^{(i)} \notin \mathcal{S}[t]^{(i)} \;\; |\mathcal{F}_t\Big) \tag{5.11}$$

$$= 1 - \mathbb{P}\Big(\bigcap_{i=1}^{N}\big\{\mathbf{x}_t^{(i)} \in \mathcal{S}[t]^{(i)} \;\; |\mathcal{F}_t\Big) \tag{5.12}$$

$$= 1 - \prod_{i=1}^{N} \underbrace{\mathbb{P}\Big(\big\{\mathbf{x}_t^{(i)} \in \mathcal{S}[t]^{(i)} \;\; |\mathcal{F}_t\Big)}_{\geq 1-\alpha-\mathcal{O}(1/T)} \tag{5.13}$$

$$\leq 1 - \prod_{i=1}^{N} \big(1 - \alpha - \mathcal{O}(1/T)\big) \tag{5.14}$$

$$= 1 - \Big(1 - \alpha - \mathcal{O}(1/T)\Big)^N \leq \gamma. \tag{5.15}$$

Since we aim for an asymptotic bound, we take $T \to \infty$ which allows us to omit the $\mathcal{O}(1/T)$ term. To obtain the smallest value of α possible, we meet the loose inequality with equality, and find α such that $1 - (1-\alpha)^N = \gamma$. Solving for α yields the value $\alpha = 1 - (1-\gamma)^{\frac{1}{N}}$. □

Metrics Computation

- Coverage rate: Letting previous notation prevail, we determine k^{th} prediction step coverage rate for any given scene by computing

$$\mathbf{1}\left\{ \bigcap_{i=1}^{N} \{\mathbf{x}_{t+k\Delta t}^{(i)} \in \mathcal{S}[t+k\Delta t]^{(i)} \right\}$$

  per timestep. We average this quantity over timesteps to obtain average coverage rate for the specific scene.

- Conservatism: Define $\mathbf{x}_t^{(E)}$ to be the state of the ego vehicle in the ground truth data at time t, and define $\widehat{\mathbf{x}}_t^{(E)}$ to be the state of the ego vehicle controlled by the planner, at time t. For any given scene, we compute

$$\frac{\min_t \min_{i\in[N]} \|\widehat{\mathbf{x}}_t^{(E)} - \mathbf{x}_t^{(i)}\|_{xy}}{\min_t \min_{i\in[N]} \|\mathbf{x}_t^{(E)} - \mathbf{x}_t^{(i)}\|_{xy}},$$

  where $\|\cdot\|_{xy}$ computes the norm only along the x and y spatial coordinates of the given state vector. We average this quantity across all scenes. Intuitively, this is a measure of how close the planner is willing to get to other vehicles, compared to the ground truth ego vehicle movement.

- Progress: Let $\mathbf{x}_g$ denote the goal state of the ego vehicle, and let $\mathbf{x}_s$ denote its starting state. Next, let $\mathbf{x}_f$ denote the final state of the ego vehicle once the planner no longer provides any new plans. We define progress as

$$1 - \frac{\|\mathbf{x}_g - \mathbf{x}_f\|_{xy}}{\|\mathbf{x}_g - \mathbf{x}_s\|_{xy}}.$$

Un-Calibrated Confidence Set Metrics, Waymo Dataset

In Tables 5.3 and 5.4, we compare the coverage rate and set sizes between the un-calibrated version of our algorithm, using only quantile regression as a confidence-set-generating tool, and the calibrated version of our algorithm, leveraging conformal prediction.

We primarily notice that conformal prediction enforces that set coverage remains above the desired 95% threshold, although this comes at the cost of providing larger sets. Additionally, conformal prediction aids in reducing the variance of coverage rates, especially for later-timestep predictions. This demonstrates the $\mathcal{O}(1/T)$ convergence guarantee on the deviation of the realized, empirical error rate from the desired error rate – with more data, the absolute deviation in conformal prediction's error rate from the desired error rate decreases, whereas this is not guaranteed for quantile regression alone.

Table 5.3: Un-Calibrated Confidence Set Coverage Metrics

Prediction Step	Calibrated Set Coverage Rate	Un-Calibrated Set Coverage Rate
0.5	0.997 ±0.002	0.968 ±0.007
1.0	0.986 ±0.006	0.887 ±0.015
1.5	0.980 ±0.008	0.839 ±0.019
2.0	0.967 ±0.011	0.817 ±0.022
2.5	0.965 ±0.011	0.818 ±0.025
3.0	0.965 ±0.013	0.800 ±0.028

Table 5.4: Un-Calibrated Confidence Set Sizes

Prediction Step	Calibrated Set Size	Un-Calibrated Set Size
0.5	61 ±6	42 ±4
1.0	246 ±27	153 ±16
1.5	655 ±71	406 ±46
2.0	1361 ±143	815 ±86
2.5	2422 ±240	1434 ±145
3.0	3885 ±365	2300 ±226

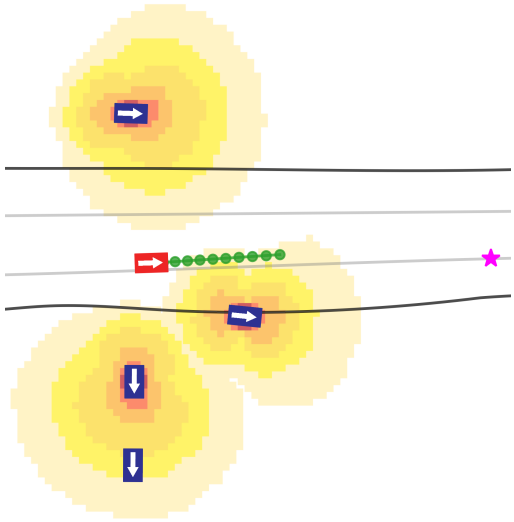


Figure 5.5: Visualization of the planner. The autonomous ego vehicle is shown in red, and the human drivers are shown in blue. The plan is shown in green, representing the tracking of the original HJ-generated plan using iLQR.

Planner Visualization

Additional Algorithm Implementation Details

We opt for the version of Trajectron++ without the encoder for maps and the encoder for future ego-agent motion plans due to the lack of availability of these in the datasets. The model architecture and hyperparameters are kept the same as in [132].

We explored graph-based planning algorithms such as A* and Dijkstra but found them to be computationally intractable as a result of the high dimensionality of the Extended Dubins' car model and the presence of time-varying dynamic obstacles. Our reachability-based planner does not suffer from such issues.

In the case in which no feasible plan to the desired target state exists, for example due to the large sizes of the probabilistic reachable sets, our reachability-based planner produces a plan minimizing the distance between the final ego state and the desired target state.

Chapter 6

Active Calibration of Reachable Sets Using Approximate Pick-to-Learn

This chapter is based on work with Ebonye Smith, Jingqi Li, and Claire J. Tomlin in [39].

6.1 Introduction

In many safety-critical applications, reachability analysis [15] is a valuable tool for providing safety guarantees for dynamical systems. Traditionally, reachability methods are computationally intensive, are performed offline, and require pre-specified models of dynamics, objectives, and constraints. Some of these limitations have been addressed by the use of learning-based algorithms to solve for reachable sets [13, 73, 99]. However, not only can learning-enabled techniques be error-prone, but also system modeling assumptions may not match reality. For example, [103] provides examples of how incomplete training can cause a learned reachable set to misclassify unsafe points as safe near its boundary. In the absence of hard safety guarantees, one approach from prior literature is to turn to *probabilistic* guarantees, and to *calibrate* a reachable set using samples [103, 104].

However, producing a strong probabilistic guarantee may require taking a large amount of potentially costly samples. Also, as the calibration procedure *adapts* the reachable set to the environment, the set updates in response to each sample that is taken. This *iterative* learning and sampling breaks assumptions made by many prior works that the samples be exchangeable or IID. Finally, there is the question of how to define calibration in the first place. Understanding how to update a reachable set is a learning problem in itself, which prior works often simplify by making the calibrated reachable set a simple function of the original set.

We address these issues by framing calibration as an active learning problem, through which we can choose only the necessary samples. At the same time, we provide strong guarantees and high-accuracy reachable sets. For this, our key insight lies in leveraging the Pick-to-Learn algorithm, introduced by Paccagnan *et al.* [126]. This is a meta-algorithm

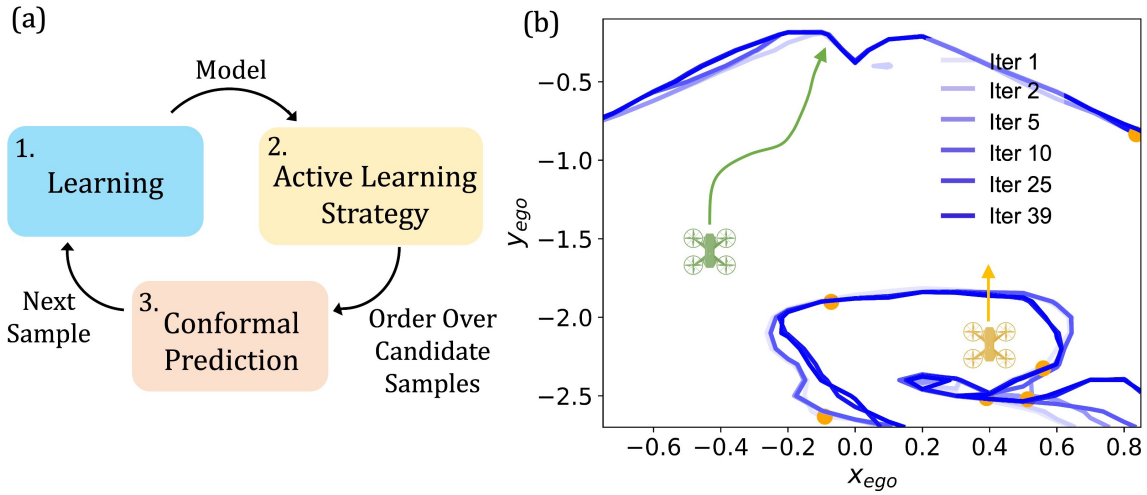


Figure 6.1: (a) Our framework provides a probabilistic generalization bound for the *iterative* process of learning and actively sampling the next data point. The novelty of our method lies in the realization that if we conformally calibrate our active sampling strategy so that it approximates maximizing the true model error, we can achieve a desired probabilistic guarantee. (b) We demonstrate our method on a drone racing example in which the ego drone (green) overtakes another drone (yellow). We display the reach-avoid set learned by our method, with the lighter colors indicative of the set at previous iterations and the darker colors for later iterations. The orange points are samples taken by our calibrated active learning strategy.

that turns any supervised learning procedure into a compression scheme, which yields a generalization bound. We adapt this algorithm for active learning, resulting in our *Approximate Pick-to-Learn* method. We overview our method in Fig. 6.1. In summary, our contributions are:

- We frame reachable set calibration as an active learning problem and provide a novel method to produce a probabilistic guarantee.
- We introduce an extension to the Pick-to-Learn algorithm for active learning in the form of Approximate Pick-to-Learn and theoretically establish its probabilistic guarantees.
- We demonstrate our methodology in simulation and show how it balances tradeoffs between sample complexity, guarantee strength, and reachable set accuracy better than the baselines.

We first give an overview of related work in Section 6.2. We then describe the problem setup in Section 6.3, explain our method in Section 6.4, show our simulation experiments in

Section 6.5, and conclude in Section 6.6.

6.2 Related Work

Calibrating Learned Reachability Computations

The works [103] and [104], which serve as our baselines, study the calibration of learned reachable sets using sampled trajectories. The iterative method [103] evaluates ground-truth reachability values at IID samples and adjusts the level set of the value function to obtain zero safety violations, while the robust method [104] instead counts violations and incorporates them into a relaxed probabilistic guarantee. Both rely on scenario optimization [28], trading off the strength of the guarantee with the sample complexity. Our work builds on these ideas with a more sample-efficient, active learning-based calibration approach.

There are also prior works on adapting reachable sets online, such as in response to new sensed information [12], with respect to changing dynamics or environmental parameters [21, 85, 120], or in accordance with human feedback about constraints [4, 134]. However, these works constrain the amount of adaptation that can happen and do not have probabilistic guarantees on the final reachable set.

Guarantees for Active and Online Learning Settings

Many works in the multi-armed bandits or Bayesian optimization literature make regret-related theoretical statements [23, 35, 145, 170], which describe the optimality of the active learning process but not how the final model performs on unseen data. They may otherwise assume that the learned model belongs to a particular hypothesis class [59, 114]. For example, some Bayesian optimization works produce probabilistic statements on the accuracy of level set estimates, but with the assumption that the underlying function is drawn from a Gaussian process (GP) [76, 81]. In contrast, we work with arbitrary and potentially black-box models of the reachability value function.

Another line of related work applies conformal prediction [9] to produce probabilistic safety guarantees in dynamic settings. For example, [49, 118] use a form of Adaptive Conformal Inference [66], whose guarantee is similar to a regret bound. Our method uses conformal prediction, but to calibrate the estimate of where it is important to sample next rather than to calibrate the reachable set itself.

6.3 Problem Definition

Let $x \in \mathcal{X}$ denote a state in the bounded state space, and let $x_{t+1} = f(x_t, u_t)$ describe the system dynamics, where $x_t \in \mathcal{X}$ and $u_t \in \mathcal{U}$ denote the state and control at time step t,

respectively. For an arbitrary policy $\pi : \mathcal{X} \to U$, the value function induced by the policy is

$$V_\pi(x,t) := \sup_{t'=0,\dots,t} \min\{\gamma^{t'} r(\xi_x^\pi(t')), \min_{\tau=0,\dots,t'} \gamma^\tau c(\xi_x^\pi(\tau))\} \tag{6.1}$$

for discount factor $\gamma \in (0,1)$, where $\xi_x^\pi : [0,t] \to \mathcal{X}$ represents the trajectory under π starting at x. Here, $r : \mathcal{X} \to \mathbb{R}$ and $c : \mathcal{X} \to \mathbb{R}$ are functions such that $r(x) > 0$ indicates that x lies in the target set, and $c(x) > 0$ indicates that x satisfies constraints. The super-zero level set of $V_\pi(x,t)$ recovers the *reach-avoid set*, which consists of all states from which the policy guides the system to the target set within t stages while satisfying the constraints.

In our experiments, we obtain a policy $\tilde{\pi}(x)$ and associated value function $\tilde{V}(x,t)$ using the RL-based method in [99]. Since the learned value function may be subject to error, we use it to initialize our active learning calibration process. Let the *hypothesis* $h : \mathcal{X} \to \mathbb{R}$, with initial hypothesis h_0, refer to $V_{\tilde{\pi}}(x,T)$ for time horizon T. During the active learning procedure, we iteratively choose x's and observe $V_{\tilde{\pi}}(x,T)$ by rolling out $\tilde{\pi}$ and applying Eq. (6.1). This provides samples $z := (x, V_{\tilde{\pi}}(x,T))$. Let z_x denote the state portion of z. We also assume access to a learning algorithm L, which maps a list of samples to a hypothesis, and an error function $e_h(z)$. In our Approximate Pick-to-Learn setting, this error is binary-valued: $e_h(z) = \mathbf{1}\{h(x) \times V_{\tilde{\pi}}(x,T) \leq 0 \ \wedge \ h(x) \neq V_{\tilde{\pi}}(x,T)\}$, so it is 0 when h correctly predicts the sign of the value and 1 otherwise. Our aim is to improve our hypothesis h while also using the collected data to produce a probabilistic guarantee.

6.4 Approximate Pick-to-Learn

In this section, we describe our adaptation of the Pick-to-Learn algorithm for active learning. We first provide an overview of Pick-to-Learn. We then describe the key ingredients that should be adapted for our algorithm: Pick-to-Learn's dataset and hypothesis-dependent total order. Finally, we describe how we use conditional conformal prediction to produce a calibrated *approximate* total order and obtain our final guarantee.

The Pick-to-Learn Algorithm

In the standard supervised learning setting, the Pick-to-Learn algorithm [126] produces a generalization bound by turning any learning algorithm into a *compression scheme*. For a dataset $D := \{z_1, ..., z_{n_D}\}$ with IID z_i's in $\mathcal{Z}$, the method aims to iteratively build the compressed set $Q \subseteq D$. In our case, $\mathcal{Z} := \mathcal{X} \times \mathbb{R}$ so each z is a combination of state and value label. At each iteration, the data point in $D \setminus Q$ on which the current hypothesis performs the worst is added to Q. This is the point that maximizes a *hypothesis-dependent total order* $\leq_h$ defined over $\mathcal{Z}$: $z_i \leq_h z_j \iff e_h(z_i) \leq e_h(z_j), \ \forall z_i, z_j \in \mathcal{Z}$. Then, L is applied to obtain a new h. The algorithm terminates when $e_h(z) \leq \omega \ \forall z \in D$ for the threshold ω and returns Q and the final hypothesis h_f.

Algorithm 1: $\mathcal{A}_C(D)$ - Approximate Pick-to-Learn for Calibrating Reachable Sets

Initialize: $Q = \emptyset$, $h = h_0$, $C = \{z_1^C, ..., z_{n_C}^C\}$, $\lambda_{h_0,C} = Quantile_{\lceil (1-\alpha)(n_C+1)\rceil / n_C}$ $\left\{ \frac{|e_{h_0}(z_1^C) - a_{h_0}(z_{1,x}^C)|}{\mu_{h_0}(z_{1,x}^C)}, ..., \frac{|e_{h_0}(z_{n_C}^C) - a_{h_0}(z_{n_C,x}^C)|}{\mu_{h_0}(z_{n_C,x}^C)} \right\}$, $\hat{e}_{h_0,C} = a_{h_0} + \lambda_{h_0,C}\mu_{h_0}$, $\bar{x} = \arg\max_{x \in D} \hat{e}_{h_0,C}(x)$

1: **while** $\hat{e}_{h,C}(\bar{x}) \geq \omega$ **do**
2: Obtain $V_{\tilde{\pi}}(\bar{x}, T)$ and define $\bar{z} := (\bar{x}, V_{\tilde{\pi}}(\bar{x}, T))$
3: $Q \leftarrow Q \cup \{\bar{z}\}$
4: $h \leftarrow L(Q)$
5: **return** h, Q, **if** $D \setminus Q = \emptyset$
6: $\lambda_{h,C} \leftarrow Quantile_{\lceil (1-\alpha)(n_C+1)\rceil / n_C}$ $\left\{ \frac{|e_h(z_1^C) - a_h(z_{1,x}^C)|}{\mu_h(z_{1,x}^C)}, ..., \frac{|e_h(z_{n_C}^C) - a_h(z_{n_C,x}^C)|}{\mu_h(z_{n_C,x}^C)} \right\}$
7: $\hat{e}_{h,C} \leftarrow a_h + \lambda_{h,C}\mu_h$
8: $\bar{x} \leftarrow \arg\max_{x \in D \setminus Q} \hat{e}_{h,C}(x)$
9: **end while**
10: **return** h, Q

Adapting Pick-to-Learn for Active Learning

As Pick-to-Learn is designed for offline supervised learning, several aspects need to be changed to apply the technique to active learning. Firstly, in active learning, a *labeled* dataset D does not exist. Because of this, we also do not have access to $e_h(z)$ for all $z \in D$, and so secondly, we do not know the hypothesis-dependent total order.

Our response to these dilemmas is to redefine D as an *unlabeled* dataset of n_D IID samples of x from a uniform distribution over the state space; $D := \{x_1, ...x_{n_D}\}$. Though now $D \subseteq \mathcal{X}$, we will allow Q to have labels so that $Q \subseteq \mathcal{Z}$. In this way, we think of D as a representation of the world in which we are performing active learning; we can choose which elements of D we would like to learn on. In the Pick-to-Learn algorithm, this choice is precisely adversarial in that it is the point for which $e_h(z)$ is maximized. However, in the absence of knowing $e_h(z)$ exactly, one can imagine designing an active learning strategy that chooses points that reduce uncertainty, optimize an objective, or follow a heuristic [64]. Generally speaking, an active learning strategy can be modeled as maximizing some function $a_h(x)$.

Calibrating the Active Learning Strategy with Conditional Conformal Prediction

In Pick-to-Learn, an upper bound is produced on the *risk* of the hypothesis, which is defined as $P_z(e_h(z) \geq \omega | D)$. However, if the termination condition of our algorithm were $a_h(x) \leq \omega \ \forall x \in D$, we would only obtain an upper bound on $P_x(a_h(x) \geq \omega | D)$. To address this issue, we leverage conformal prediction with calibration set C to produce a value $\hat{e}_{h,C}(x)$, serving as an adjusted version of $a_h(x)$, so that with high probability, $e_h(z) \geq \omega$ implies that

$\hat{e}_{h,C}(z_x) \geq \omega$ for any z_x and h.

We detail our method in Algorithm 1. Following one technique given in [9], we construct an interval around $a_h(x)$. Given a heuristic value $\mu_h(x)$ that serves as an estimate of the distance between a_h and e_h, conditional conformal prediction calculates $\lambda_{h,C} \in \mathbb{R}$ so that:

$$\begin{aligned} P_C[1-\alpha-\epsilon_\alpha \leq P_z[e_h(z) \in [a_h(z_x) - \lambda_{h,C}\mu_h(z_x), \\ a_{h,\eta}(z_x) + \lambda_{h,C}\mu_{h,}(z_x) \,|\, C, D] \leq 1-\alpha+\epsilon_\alpha \,|\, D] \geq 1-\beta \end{aligned} \tag{6.2}$$

for calibration set $C := \{z_1^C, ..., z_{n_C}^C\}$ containing IID state samples separate from D. In essence, C is a set of state samples for which we have value labels and therefore know $e_h(z)$ for calibration purposes. $\lambda_{h,C}$ is the $\lceil (1-\alpha)(n_C+1) \rceil / n_C$ quantile of $\{s(z_1^C), ..., s(z_{n_C}^C)\}$ for score function $s(z) := |e_h(z) - a_h(z_x)|/\mu_h(z_x)$ (Line 6 in Algorithm 1). From [9], the inner probability in Equation (6.2) follows a Beta distribution. For some constant β, the size of C is calculated such that $1-\beta$ probability mass of the distribution remains within $1-\alpha-\epsilon_\alpha$ and $1-\alpha+\epsilon_\alpha$ for tolerance ϵ_α.

Now, we write $\hat{e}_{h,C}(x) := a_h(x) + \lambda_{h,C}\mu_h(x)$ (Line 7). This produces a total order $\hat{\leq}_h$ that *approximates* $\leq_h$ over z_x: $x_i \;\hat{\leq}_h\; x_j \iff \hat{e}_{h,C}(x_i) \leq \hat{e}_{h,C}(x_j),\ \forall x_i, x_j \in \mathcal{X}$. We impose that $\mu_h(x)$ must never be 0. The Pick-to-Learn algorithm can now be performed as is except by maximizing $\hat{e}_{h,C}(x)$ at every step (Line 8) and with the termination condition $\hat{e}_{h,C}(x) \leq \omega \; \forall x \in D$ (Line 1). We present our final guarantee with the following theorem:

Theorem 1. *Let $\mathcal{A}_C$ be Algorithm 1, such that $\mathcal{A}_C(D) = (h_f, Q)$. Pick $\alpha, \epsilon_\alpha, \beta, \delta \in (0,1)$. Then,*

$$P_{C,D}[P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq \bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha] \geq 1 - \beta - \delta,$$

where $\bar{\epsilon}(n, \delta)$ for $n = |Q|$ is the unique solution of $\Psi_{n,\delta}(\epsilon) = \frac{\delta}{2n_D} \sum_{m=n}^{n_D-1} \frac{\binom{m}{n}}{\binom{n_D}{n}} (1-\epsilon)^{-(n_D-m)} + \frac{\delta}{6n_D} \sum_{m=n_D+1}^{4n_D} \frac{\binom{m}{n}}{\binom{n_D}{n}} (1-\epsilon)^{m-n_D} = 1$ in the interval $[n/n_D, 1]$ and $\bar{\epsilon}(n_D, \delta) = 1$.

Proof. We provide a proof sketch here and defer the detailed proof to the Appendix at the end of this chapter. We use Theorem 4 from Campi and Garatti (2023) [27] to probabilistically bound a quantity called the *probability of change of compression*. The theorem can be used to bound the probability that the error of the hypothesis is high ($e_{h_f}(z) \geq \omega$) if the event of the error being high is related to the event of the change of compression. Thus, we take the following general steps: we first establish that with high probability over C, it is unlikely that $e_h(z) \geq \omega$ but $\hat{e}_{h,C}(z_x) \leq \omega$. We then relate the event $e_h(z) \leq \omega \vee \hat{e}_{h,C}(z_x) \geq \omega$ to the fact that $\mathcal{A}_C$ would produce a different output Q were it given that additional z (change of compression). Then, we can combine various probabilistic statements to arrive at our final statement. □

Theorem 1 suggests that, with high probability, the learned value function has low error on unseen states even though the samples used for verification came from an adaptive process.

Remark 1. *Although we have designed a deterministic algorithm, we would like to allow for practitioners to incorporate exploration in choosing x by allowing $a_h(x)$ to be a function of a random variable. This would mean that $\mathcal{A}_C$ should be seeded in practice so that it outputs the same quantities for the same random seed. Indeed, we implement this technique in our experiments. Along the same vein, since $\hat{e}_{h,C}$ should induce a total order, then if there are ties, where $\hat{e}_{h,C}(x_i) = \hat{e}_{h,C}(x_j)$ for $x_i \neq x_j$, a tie-breaking criterion can be implemented using a small amount of random noise: $\hat{e}_{h,C}(x) := a_h(x) + \lambda\mu_h(x) + \sigma(x)$ for positive-valued $\sigma(x)$ such that $\hat{e}_{h,C}(x) \geq a_h(x) + \lambda\mu_h(x)\ \forall x \in \mathcal{X}$ still holds. Note that for our algorithm, we only impose that $\hat{e}_{h,C}$ induces a total order and not that e_h induces a total order, in contrast with Pick-to-Learn.*

6.5 Simulation Experiments

Experimental Setup. We consider the example in [99] of an ego drone competing with another drone to reach a gate. For this reachability problem, the state is 12-dimensional, consisting of the two agents' 6-dimensional dynamics that are described by double integrators in the three spatial dimensions. The reward function $r(x)$ incentivizes the ego drone to remain ahead of the other drone, move faster than the other drone, and approach the gate within a corridor. The constraint function $c(x)$ imposes that the ego drone approach the gate within a certain angle, maintain an altitude limit, and stay away from the other agent and its approximate region of downwash. We produce $V_{\tilde{\pi}}(x, T)$ values by applying $\tilde{\pi}$ for the ego drone and applying a PID controller for the other drone for $T = 30$ steps.

In order to avoid retraining a neural network at every iteration, we model $h_0(x)$ as a GP that uses $\tilde{V}(x, t)$ as part of its prior. Thus, the total sample complexity of our method for these experiments is $|Q| + |C|$ plus 40 initial samples for fitting $h_0(x)$. We opt for a heuristic active learning strategy in which we prioritize points near the boundary of the reachable set, where it is most likely that the model is incorrect. Since the zero-level set of the value function defines the boundary, $a_h(x)$ simply uses the absolute value of the value function and is normalized so that $a_h(x) \in [0, 1]$: $a_h(x) := 1 - |h(x)|/(\max_x |h(x)|)$.

Remark 2. *We are able to keep a low sample complexity by sampling C once and using it throughout the algorithm. The dependency between C and h is quite weak since h is learned based on only samples from D. Our empirical findings validate the theoretical results of Theorem 1.*

Note that our algorithm terminates when $\hat{e}_{h,C}(x) \leq \omega\ \forall x \in D$, and since $\hat{e}_{h,C}(x) \geq a_h(x)$, if $a_h(x)$ is never below $\omega\ \forall x \in D$ during the course of the algorithm then neither is $\hat{e}_{h,C}(x)$. This is indeed the case for our particular strategy $a_h(x)$. To remedy this, we adopt the engineering solution of decaying $a_h(x)$ at every iteration of the while loop in Algorithm 1 before the conformal prediction step is applied using a multiplicative exponential factor $\zeta \in [0, 1]$, thus maintaining the guarantee. Also, we heuristically define $\mu_h(x)$ as $0.1 \times a_h(x)$. The value of $\hat{e}_{h,C}(x)$ at every iteration depends on the values of the score functions in the calibration

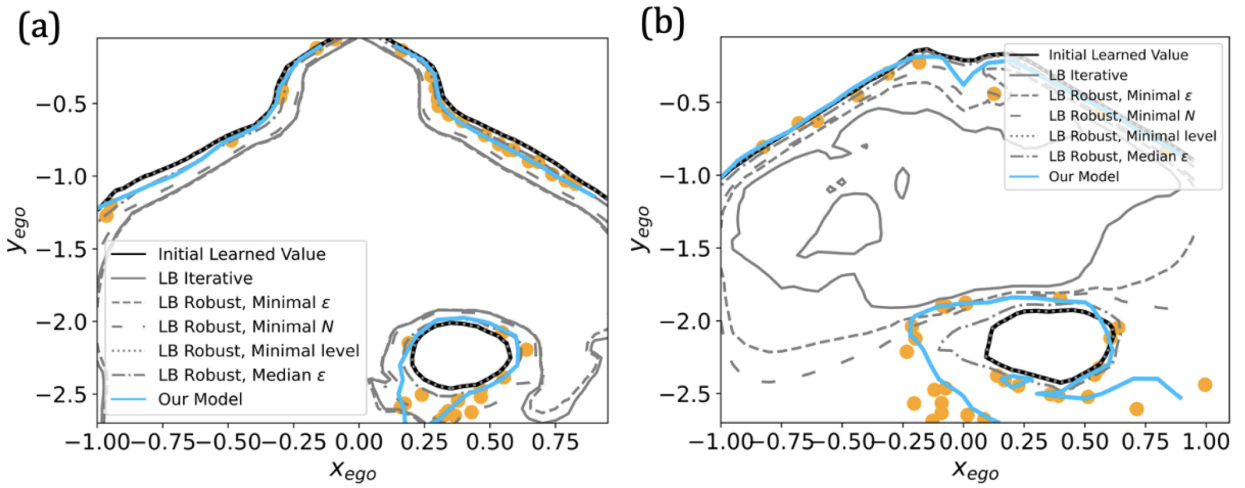


Figure 6.2: (a) In this 2-dimensional "Slice 1", the ego drone's 3D velocity is set to $[0.0, 0.7, 0.0]$ and its altitude is 0.0. The other drone's 3D spatial coordinates are $[0.4, -2.2, 0.0]$ and its 3D velocity is $[0.0, 0.3, 0.0]$. (b) In this 2-dimensional "Slice 2", the other drone's state is the same, but the ego drone's velocity is $[0.0, 0.0, -0.5]$ and its altitude is 0.05. The baselines' calibration technique is to simply choose an appropriate level of the learned value function. For the more commonly seen scenario (a), the level sets have more regular shapes, better justifying this calibration technique. However, for the less commonly seen scenario (b), this is not the case, and it is more reasonable to be unconstrained by learned set geometries, as in our method. In both figures, orange points are the samples that our method took.

dataset. If for some $z_i^C \in C$ $a_h(z_{i,x}^C)$ is small but $e_h(z_i^C) = 1$, $\hat{e}_{h,C}(x)$ may remain above ω for some $x \in D$. It is only when discrepancies between a_h and e_h lessen, usually by values of e_h in C going to zero, that the algorithm is allowed to terminate. We set $\omega = 0.3$, $\zeta = 0.95$, and $\delta = 1e^{-4}$.

Baselines and Metrics. We compare our method with the two baselines described in Section 6.2: [103], which we call LB Iterative after the names of the authors, and [104], which we denote as LB Robust. Both guarantee $P_{x \in \hat{\mathcal{S}}}(V_{\tilde{\pi}}(x, T) \leq 0) \leq \epsilon_{LB}$ with probability at least $1 - \beta$ over sampled data, where $\hat{\mathcal{S}}$ is the calibrated set. We set $\beta = 0.1$ for all methods. Notably, our probabilistic guarantee is with respect to a general notion of error, whereas the baselines' guarantee only constrains the false inclusion of unsafe states.

We implement the LB Robust method to highlight the trade-offs involved in calibration. We vary the number of samples over ($[50, 200, 250, 300, 350, 500, 750, 1000]$) and the levels of the learned value function ($[0.0, 0.05, 0.1, 0.15, 0.2, 0.3, 0.5, 0.75, 0.9, 1.0]$). For each combination of these, we implement the method to obtain the ϵ_{LB} value. Then, over all combinations

we take the one with the minimal ϵ_{LB} ("LB Robust, Minimal ϵ"), the minimal ϵ_{LB} for the smallest number of samples ("LB Robust, Minimal N"), the minimal ϵ_{LB} for the smallest level ("LB Robust, Minimal level"), and finally the median ϵ_{LB} ("LB Robust, Median ϵ").

For all methods, we compare the total number of samples required, ϵ_{LB} and $\bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha$, and the false positive rate and false negative rate (FPR and FNR) of the final sets. Here, FPR and FNR are calculated by gridding the state space, computing the ground truth values at the grid points, and checking whether the calibrated set includes (a "positive") and excludes (a "negative") the correct points.

A link to our code is at `https://github.com/sdeglurkar/bayes_opt_calibration`.

Results and Discussion. We vary the type of active learning technique we use, the dimension and slice of the state space that we operate in, and the values of α and ϵ_α. Fig. 6.2 visualizes the impact of some of these variations in 2-dimensional slices of the 12-dimensional system. Compared to Fig. 6.2(a), the states in the slice in Fig. 6.2(b) are less likely to have been seen during the learning of $\tilde{V}(x, t)$. Since the baseline methods can only choose among the levels of the learned value function, they struggle more when the learned function is less accurate. Our method, meanwhile, is not constrained by the geometries of the learned level sets.

Fig. 6.3 provides comparisons of the methods for different variable combinations. Each experiment is run for 10 random seeds, and comparisons are only made for successful seeds. Here, failure is defined as more than 70 iterations of our algorithm, since the method is then no longer adaptive with minimal samples. Our method uses the least number of samples except for the "LB Robust, Minimal N" method, which has a worse FNR. As seen in Table 6.1, the tightest probabilistic bound is achieved by the "LB Robust, Minimal ϵ" method, although it also uses the most samples. Combining conformal prediction with Pick-to-Learn in our work weakens our guarantee in comparison. However, we emphasize that this is because our method incorporates *learning and adaptation* with respect to any error function, which is necessary for many real-world scenarios. Our method also more effectively trades off between FNR and FPR compared with the baselines, which may minimize one at the expense of the other.

Across our method variants, boundary and random active learning require a similar number of samples. Though seemingly counterintuitive, this reflects a key property of probabilistic guarantees. The required number of iterations depends on the particular calibration set C and how well the model learns per sample. Especially if α and ϵ_α are larger values, the $\hat{e}_{h,C}(x) \leq \omega$ condition may not be as difficult to satisfy within a set number of iterations. However, even if the Boundary and Random experiments yield a similar probabilistic guarantee due to using a similar number of samples, the final reachable set learned via the boundary sampling strategy tends to have a better FPR and FNR.

Finally, we study how our method's performance scales with increasing state dimension, where model learning becomes more difficult and reducing errors in C becomes harder. In these cases, relaxing α and ϵ_α reduces the required number of samples, at the cost of higher FPR and FNR. We use $(\alpha, \epsilon_\alpha) = (0.05, 0.03)$ for the 2D experiments ($|C| = 112$,

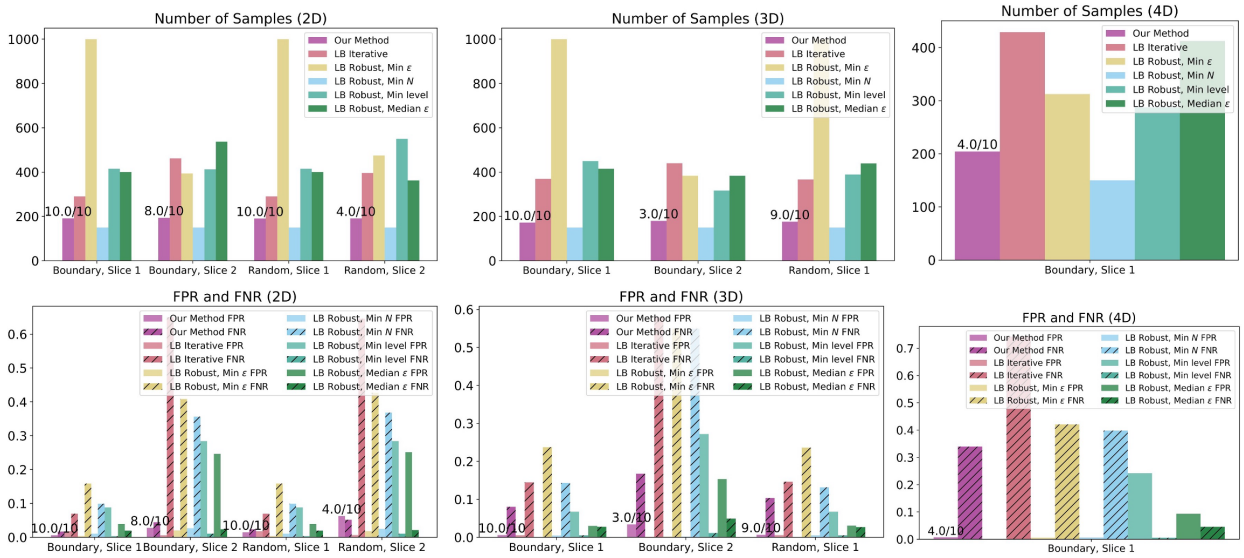


Figure 6.3: Plots comparing the number of samples and FPR/FNR across methods. "Boundary" denotes our boundary sampling active learning technique, while "Random" refers to setting $a_h(x)$ to random values. In 3D experiments, the slice of the dynamics is the same as that for the 2D Slice 1 except the ego drone's altitude is also allowed to vary. In 4D experiments, we additionally allow the ego agent's velocity in the x-direction to vary. The numbers above the bars indicate how many seeds out of 10 were successful for our method. Our method balances the competing objectives of minimizing the number of samples and FPR and FNR well compared with the baselines. As we expect, performance for all methods is worse for Slice 2 compared with Slice 1.

$|D| = 4000$), $(0.1, 0.05)$ for the 3D experiments ($|C| = 93$, $|D| = 4000$), and $(0.15, 0.05)$ for the 4D experiments ($|C| = 112$, $|D| = 8000$). By uniting the processes of learning and probabilistic assurance, our framework allows for a formal examination of how they interact and trade off.

6.6 Conclusions

In this work, we (1) introduce a method to use active learning to both use fewer samples and gain more accuracy when calibrating reachable sets and (2) produce an extension to the Pick-to-Learn algorithm that allows us to provide a probabilistic guarantee for active learning. In this way, our work allows for an adaptive notion of safety. We believe that our method can be used not only for calibration but also for reachable set synthesis more generally. In our next steps, we would like to apply these principles in hardware experiments. From our analyses, we additionally see that our intuitive understanding of risk in dynamical systems does not

Table 6.1: Comparing $\bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha$ and ϵ_{LB} values to understand the tightness of the probabilistic bounds - lower values generate stronger statements. "LB Robust, Min ϵ" always has the lowest ϵ, but it uses the most samples.

Setting	Ours	LB Iterative	LB Robust Min ϵ	LB Robust Min N	LB Robust Min Level	LB Robust Median ϵ
2D						
Boundary, Slice 1	0.099	0.05	0.003	0.016	0.035	0.024
Boundary, Slice 2	0.1	0.05	0.01	0.021	0.093	0.094
Random, Slice 1	0.099	0.05	0.003	0.016	0.035	0.024
Random, Slice 2	0.099	0.05	0.009	0.019	0.096	0.095
3D						
Boundary, Slice 1	0.169	0.05	0.003	0.016	0.040	0.028
Boundary, Slice 2	0.172	0.05	0.007	0.016	0.160	0.118
Random, Slice 1	0.171	0.05	0.003	0.016	0.039	0.028
4D						
Boundary, Slice 1	0.212	0.05	0.015	0.026	0.203	0.111

always align with the probabilistic statements made by statistical techniques. For example, the size of C prescribed by conformal prediction theory is not dependent on the dimension of the state space even though it is intended to capture model error. Also, the relationship between probabilistic guarantees, set conservatism, and learning difficulty warrants further investigation in future work.

6.7 Appendix

In this Appendix, we provide a full proof of Theorem 1. In order to do so, we introduce several Lemmas. All theoretical statements are provided below.

Theoretical Statements

Lemma 1. *Let κ_C be our compression function; that is, $\kappa_C(D) := Q$. Then, κ_C is a preferent compression function given C.*

Lemma 2. *For any h present during the execution of $\mathcal{A}_C$,*
$P_C[P_z(e_h(z) \geq \omega \wedge \hat{e}_{h,C}(z_x) \leq \omega \mid C, D) \leq \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta$ for the threshold ω.

Lemma 3. $e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D) \implies e_{h_f}(z) \geq \omega \wedge \hat{e}_{h_f,C}(z_x) \leq \omega$.

Lemma 4. *Given logical statements $A(C, D)$ and $B(C, D)$ for random variables C and D, if $P_C(A(C, D)|D) \geq 1 - \beta$ and $P_D(B(C, D)|C) \geq 1 - \delta$, then $P_{C,D}(A(C, D) \wedge B(C, D)) \geq 1 - \beta - \delta$.*

Theorem 1. Let $\mathcal{A}_C$ be Algorithm 1, such that $\mathcal{A}_C(D) = (h_f, Q)$. Pick $\alpha, \epsilon_\alpha, \beta, \delta \in (0, 1)$ and calculate $\bar{\epsilon}(|Q|, \delta)$ as in Pick-to-Learn. Then,

$$P_{C,D}[P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq \bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha] \geq 1 - \beta - \delta.$$

Proof of Lemma 1

Lemma 1. Let κ_C be our compression function; that is, $\kappa_C(D) := Q$. Then, κ_C is a preferent compression function given C.

Proof. If C is given, the proof of this statement follows the proof of Lemma A.6 from Paccagnan et al. (2023) [126] since $\hat{e}_{h,C}$ defines a total order. The proof shows that, given that $\kappa_C(D \cup \{z\}) \neq \kappa_C(D)$, that $\kappa_C(D \cup \{z\}) \neq V$ for any $V \subseteq D$. □

Proof of Lemma 2

Lemma 2. For any h present during the execution of $\mathcal{A}_C$,
$P_C[P_z(e_h(z) \geq \omega \wedge \hat{e}_{h,C}(z_x) \leq \omega \mid C, D) \leq \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta$ for the threshold ω.

Proof. For all h present during the execution of $\mathcal{A}_C$, our methodology guarantees, via conditional conformal prediction, that

$$P_C[1 - \alpha - \epsilon_\alpha \leq P_z(e_h(z) \in [a_h(z_x) - \lambda_{h,C}\mu_{h,\eta}(z_x),$$
$$a_h(z_x) + \lambda_{h,C}\mu_h(z_x)] \mid C, D) \leq 1 - \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta$$

for suitable values of $\alpha, \epsilon_\alpha, \beta$, and $|C|$, for $\lambda_{h,C}$ computed via conformal prediction, for any z selected independently from C and D, and for any arbitrary $a_h(x)$ and $\mu_h(x)$. We impose that $\mu_h(x)$ must never be zero. We write $\hat{e}_{h,C}(x) := a_h(x) + \lambda_{h,C}\mu_h(x)$.

Rewriting the guarantee from conformal prediction, we have

$$P_C[1-\alpha-\epsilon_\alpha \leq P_z(a_h(z_x) - \lambda_{h,C}\mu_h(z_x) \leq e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x) \mid C, D) \leq 1-\alpha+\epsilon_\alpha \mid D] \geq 1-\beta$$

Since $1-\alpha-\epsilon_\alpha \leq P_z(a_h(z_x)-\lambda_{h,C}\mu_h(z_x) \leq e_h(z) \leq a_h(z_x)+\lambda_{h,C}\mu_h(z_x) \mid C, D) \leq 1-\alpha+\epsilon_\alpha$ implies $1-\alpha-\epsilon_\alpha \leq P_z(a_h(z_x) - \lambda_{h,C}\mu_h(z_x) \leq e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x) \mid C, D)$, we have

$$P_C[1-\alpha-\epsilon_\alpha \leq P_z(a_h(z_x) - \lambda_{h,C}\mu_h(z_x) \leq e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x) \mid C, D) \mid D] \geq 1-\beta$$

Since $a_h(z_x) - \lambda_{h,C}\mu_h(z_x) \leq e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x)$ implies $e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x)$,

$$P_z(e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x) \mid C, D) \geq P_z(a_h(z_x) - \lambda_{h,C}\mu_h(z_x) \leq e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x) \mid C, D)$$

is always true. So,

$$P_C[1-\alpha-\epsilon_\alpha \leq P_z(e_h(z) \leq a_h(z_x) + \lambda_{h,C}\mu_h(z_x) \mid C, D) \mid D] \geq 1-\beta$$

and substituting $\hat{e}_{h,C}$,

$$P_C[1-\alpha-\epsilon_\alpha \leq P_z(e_h(z) \leq \hat{e}_{h,C}(z_x) \mid C, D) \mid D] \geq 1-\beta$$

This means that

$$P_C[P_z(e_h(z) \geq \hat{e}_{h,C}(z_x) \mid C, D) \leq \alpha+\epsilon_\alpha \mid D] \geq 1-\beta$$

Now we use the Law of Total Probability for the inner probability $P_z(e_h(z) \geq \hat{e}_{h,C}(z_x) \mid C, D)$:

$$P_z(e_h(z) \geq \hat{e}_{h,C}(z_x) \mid C, D) = P_z(e_h(z) \geq \omega \wedge e_h(z) \geq \hat{e}_{h,C}(z_x) \geq \omega \mid C, D) + P_z(e_h(z) \geq \omega \wedge \hat{e}_{h,C}(z_x) \leq \omega \mid C, D) + P_z(e_h(z) \leq \omega \wedge \hat{e}_{h,C}(z_x) \leq e_h(z) \leq \omega \mid C, D)$$

since ω is a fixed deterministic value.

So, $P_z(e_h(z) \geq \hat{e}_{h,C}(z_x) \mid C, D) \leq \alpha+\epsilon_\alpha$ implies that $P_z(e_h(z) \geq \omega \wedge \hat{e}_{h,C}(z_x) \leq \omega \mid C, D) \leq \alpha+\epsilon_\alpha$. This gives

$$P_C[P_z(e_h(z) \geq \omega \wedge \hat{e}_{h,C}(z_x) \leq \omega \mid C, D) \leq \alpha+\epsilon_\alpha] \geq P_C[P_z(e_h(z) \geq \hat{e}_{h,C}(z_x) \mid C, D) \leq \alpha+\epsilon_\alpha \mid D] \geq 1-\beta$$

and

$$P_C[P_z(e_h(z) \geq \omega \wedge \hat{e}_{h,C}(z_x) \leq \omega \mid C, D) \leq \alpha+\epsilon_\alpha \mid D] \geq 1-\beta.$$

□

Proof of Lemma 3

Lemma 3. $e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D) \implies e_{h_f}(z) \geq \omega \wedge \hat{e}_{h_f,C}(z_x) \leq \omega$.

Proof. We will show that $\kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D) \implies \hat{e}_{h_f,C}(z_x) \leq \omega$. Then, the statement follows.

If z is such that $\kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D)$, or $\kappa_C(Q \cup \{z\}) = Q$, then $\mathcal{A}_C(Q \cup \{z\})$ must select all of the same points as $\mathcal{A}_C(D)$, choosing the same points at every iteration. It must terminate having produced Q and h_f. This must mean that in its last iteration, $\arg\max_{\tilde{z} \in Q \cup \{z\} \setminus Q} \hat{e}_{h_f,C}(\tilde{z}_x) \leq \omega$, which implies that $\arg\max_{\tilde{z} \in \{z\}} \hat{e}_{h_f,C}(\tilde{z}_x) \leq \omega$. Then, $\hat{e}_{h_f,C}(z_x) \leq \omega$. □

Proof of Lemma 4

Lemma 4. Given logical statements $A(C, D)$ and $B(C, D)$ for random variables C and D, if $P_C(A(C, D)|D) \geq 1 - \beta$ and $P_D(B(C, D)|C) \geq 1 - \delta$, then $P_{C,D}(A(C, D) \wedge B(C, D)) \geq 1 - \beta - \delta$.

Proof. We have

$$P_C(A(C, D)|D) \geq 1 - \beta \implies$$
$$P_C(\neg A(C, D)|D) \leq \beta$$

Since

$$P_{C,D}(\neg A(C, D)) = \mathbb{E}_D[P_C(\neg A(C, D)|D)]$$

and $P_C(\neg A(C, D)|D) \leq \beta$, it follows that $\mathbb{E}_D[P_C(\neg A(C, D)|D)] \leq \mathbb{E}[\beta] = \beta$. So, $P_{C,D}(\neg A(C, D)) \leq \beta$. The reasoning is similar for $P_{C,D}(\neg B(C, D)) \leq \delta$.

Now using the union bound,

$$P_{C,D}(\neg A(C, D) \vee \neg B(C, D)) \leq \beta + \delta$$

and

$$P_{C,D}(A(C, D) \wedge B(C, D)) \geq 1 - \beta - \delta.$$

□

Proof of Theorem 1

Theorem 1. Let $\mathcal{A}_C$ be Algorithm 1, such that $\mathcal{A}_C(D) = (h_f, Q)$. Pick $\alpha, \epsilon_\alpha, \beta, \delta \in (0,1)$ and calculate $\bar{\epsilon}(|Q|, \delta)$ as in Pick-to-Learn [126]. Then,

$$P_{C,D}[P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq \bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha] \geq 1 - \beta - \delta.$$

Proof. **Lemma 2** implies that $P_C[P_z(e_{h_f}(z) \geq \omega \wedge \hat{e}_{h_f,C}(z_x) \leq \omega \mid C, D) \leq \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta$. **Lemma 3** implies that

$$P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D)) \mid C, D) \leq P_z(e_{h_f}(z) \geq \omega \wedge \hat{e}_{h_f,C}(z_x) \leq \omega \mid C, D)$$

is always true. So,

$$P_C[P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D)) \mid C, D) \leq \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta$$

Now, we use the Law of Total Probability on the inner probability to write:

$$\begin{gathered} P_z(e_{h_f}(z) \geq \omega \mid C, D) = P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D) \mid C, D) + \\ P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) \end{gathered}$$

We know that

$$\begin{gathered} P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) \leq \\ P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) \end{gathered}$$

by the monotonicity of probabilities. So, $P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D)) \mid C, D) \leq \alpha + \epsilon_\alpha$ implies $P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) + \alpha + \epsilon_\alpha$ always. So we can write

$$\begin{gathered} P_C[P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) + \alpha + \epsilon_\alpha \mid D] \geq \\ P_C[P_z(e_{h_f}(z) \geq \omega \wedge \kappa_C(\kappa_C(D) \cup \{z\}) = \kappa_C(D)) \mid C, D) \leq \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta \end{gathered}$$

or simply

$$P_C[P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) + \alpha + \epsilon_\alpha \mid D] \geq 1 - \beta.$$

Theorem 7 of [27] states that, if κ_C satisfies the property of preference, then

$$P_D[P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) | D) \leq \bar{\epsilon}(|Q|, \delta)] \geq 1 - \delta$$

We claim that this is equivalent to

$$P_D[P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) | D, C) \leq \bar{\epsilon}(|Q|, \delta) \mid C] \geq 1 - \delta$$

and will explain this reasoning at the end of this proof. We invoke **Lemma 1** (our compression function is preferent) to use this theorem.

For clarity, let us shorthand the statement $P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) \mid C, D) + \alpha + \epsilon_\alpha$ as $A(C, D)$ and the statement $P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D) | D, C) \leq \epsilon(|Q|, \delta)$ as $B(C, D)$. So, our logic so far has produced the statements

$$P_C(A(C, D)|D) \geq 1 - \beta$$
$$P_D(B(C, D)|C) \geq 1 - \delta$$

$A(C, D) \wedge B(C, D)$ is equal to

$$P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq \bar{\epsilon}(|Q|, \delta) + \alpha + \epsilon_\alpha$$

$A(C, D)$ is true with a probability over C (though for all D), and $B(C, D)$ is true with a probability over D (though for all C). Thus, we can use **Lemma 4** to merge the two statements and give our final statement:

$$P_{C,D}[P_z(e_{h_f}(z) \geq \omega \mid C, D) \leq \epsilon(|Q|, \delta) + \alpha + \bar{\epsilon}_\alpha] \geq 1 - \beta - \delta.$$

Now it remains to explain why we can write Theorem 7 in [27] as

$$P_D[P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D)|D, C) \leq \bar{\epsilon}(|Q|, \delta) \mid C] \geq 1 - \delta.$$

We argue that the original theorem did not consider a compression function that is a function of any random variable other than D, while in our case κ is also dependent on C. This means that in our case, the probability of change of compression is a function of D and C and not just D. Therefore, the probability of change of compression should be written as $P_z(\kappa_C(\kappa_C(D) \cup \{z\}) \neq \kappa_C(D)|D, C)$ for correctness. The reasoning in the proof of Theorem 4 in [27] about the distribution of the probability of change of compression with respect to D does not change. One can think of our proof as adding reasoning on the distribution of the probability of change of compression with respect to C *in addition to* D.

□

Chapter 7

Hierarchical Probabilistic Verification for Reachability Learning

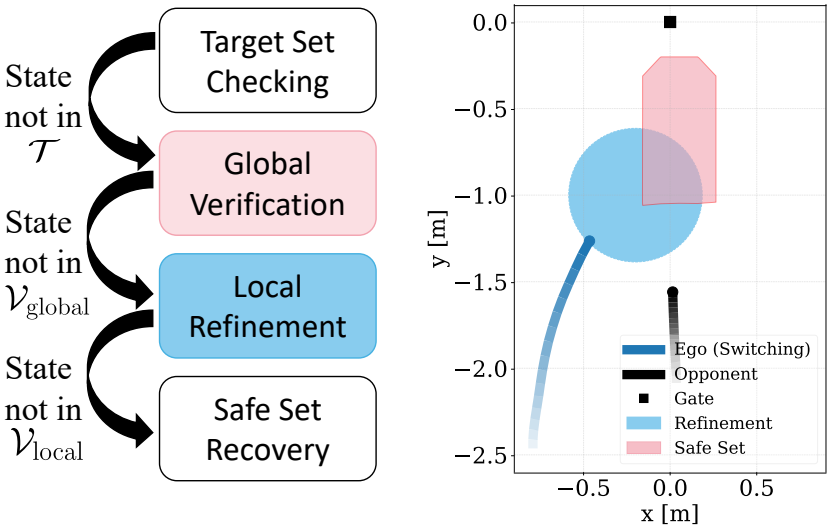
This chapter is based on work with Ebonye Smith, Jingqi Li, Gechen Qu, and Claire J. Tomlin in [142].

7.1 Introduction

Provably safe controller design is essential for deploying autonomous systems in safety-critical settings such as multi-agent coordination, human–robot interaction, and high-speed navigation in cluttered environments. Among those applications, a central challenge is to characterize the set of states from which a system can be driven to a desired goal while satisfying safety constraints [155]. Hamilton–Jacobi (HJ) reachability [112, 116, 149] computes such sets, but classical HJ methods suffer from the curse of dimensionality [15]. To overcome this issue, prior works have proposed learning-based approaches that approximate reachable sets and synthesize control policies [13, 57, 73, 99]. While these approaches enable scalability, they may lead to overly optimistic approximations that misclassify unsafe regions as safe.

To improve reliability under learning-based approximations, verification is necessary. Existing methods can be categorized along three axes. First, deterministic approaches provide worst-case guarantees but can be overly conservative in practice [1, 6, 37, 74, 98, 99], whereas probabilistic methods are typically less conservative but may require many samples to achieve tight guarantees [29, 43, 44, 46, 102, 103, 104, 105, 107]. Second, verification acting *globally* over the entire state space is computationally expensive and sensitive to outliers [105], while *local* verification focuses on regions of interest, enabling more sample-efficient certification. Third, due to its high computational cost, global verification is typically performed *offline* [103], whereas *online* refinement enables adaptive correction of overly conservative certificates during execution. Existing approaches typically operate within a single regime, limiting their ability to balance accuracy, efficiency, and scalability in high-dimensional systems.

In this work, we propose a *hierarchical* certification-guided framework, shown in Fig. 7.1,



(a) Hybrid Switching Framework

(b) Certified Global Set and Local Refinement

Figure 7.1: (a) Building on a learned reachability value function and policy from prior work (e.g., [99]), our hierarchical certification-guided framework uses probabilistically certified global and local safe sets to guide decision-making, invoking *safe set recovery* to drive the agent to a certified region. Our key novelty is leveraging scenario optimization for local safe set expansion. (b) Drone racing example in which the ego drone (blue) reaches a refined locally certified safe set which empirically results in safe overtaking of an opponent (black).

that couples global probabilistic certification with online local refinement for learning-based reachability. A coarse global certified region is first constructed via scenario optimization [28], yielding a conservative but efficient probabilistic under-approximation of the reach-avoid set under the sampling distribution. An online local refinement module then performs targeted sampling near the global certificate boundary, expanding the certified region by recovering states excluded by the conservative global approximation. The certified regions serve as decision-making guides: when the system lies within a certified region, the learned reachability policy π_{lp} is trusted; otherwise, control transfers to a model-based fallback warm-started by π_{lp}. Our contributions are:

- **Hierarchical certification-guided control framework**: A multi-tier framework combining a global probabilistic certificate with online local refinement with condition-based switching, using certified regions to guide when π_{lp} can be trusted versus deferring to a fallback controller.
- **Boundary-anchored local refinement of the probabilistically certified set:** An online scenario-based procedure that recovers probabilistically validated states near the global certificate boundary.

- **Scenario-based certification guarantees:** Theoretical guarantees characterizing the violation probability of the global and locally refined certified regions under the corresponding sampling distributions.
- **Empirical validation:** Simulation results in a 12D drone racing case study demonstrating improved success rates in safe overtaking compared to baseline methods.

7.2 Preliminaries

In this section, we review reachability analysis and scenario optimization, which underpin our new probabilistic verification framework in Section 7.3.

Reachability Learning

Consider a discrete-time nonlinear dynamical system $x_{t+1} = f(x_t, u_t)$, where $x_t \in \mathcal{X} \subseteq \mathbb{R}^n$ and $u_t \in \mathcal{U} \subseteq \mathbb{R}^m$ represent the state and control at time t. Let $r : \mathbb{R}^n \to \mathbb{R}$ be a Lipschitz-continuous reward function with Lipschitz constant L_r, where $r(x) > 0$ indicates that a state x is in the target set $\mathcal{T}$. Let $c : \mathbb{R}^n \to \mathbb{R}$ be a Lipschitz-continuous constraint function with Lipschitz constant L_c, where $c(x) \leq 0$ indicates constraint violation. Our goal is to identify the set of initial states from which the system can be safely controlled to $\mathcal{T}$ by policy $\pi : \mathbb{R}^n \to \mathcal{U}$ within a finite time horizon $T \in \mathbb{Z}_+$, referred to as the *reach-avoid (RA)* set [113]:

$$\mathcal{R} := \left\{ \begin{array}{l} x_0 : \exists \pi \text{ such that } \exists t < T, \\ (r(x_t) > 0 \ \wedge \ \forall \tau \in [0, t], c(x_\tau) > 0) \end{array} \right\}. \tag{7.1}$$

Denote by $\xi^\pi_{x_0}$ a state trajectory from a state x_0 under control policy π. We define the time-discounted *RA measure* $g_\gamma(\xi^\pi_{x_0}, t)$ as

$$g_\gamma(\xi^\pi_{x_0}, t) := \min \Big\{ \gamma^t r(x_t), \min_{\tau=0,\ldots,t} \gamma^\tau c(x_\tau) \Big\}, \tag{7.2}$$

where $\gamma \in (0, 1)$ and we have $g_\gamma(\xi^\pi_{x_0}, t) > 0$ if and only if there exists a trajectory from x_0 reaching the target set safely at time step t.

Given a horizon T, the finite-horizon *RA value function* $V_\gamma(x, T)$ evaluates the maximum RA measure:

$$V_\gamma(x_0, T) := \max_\pi \max_{t=0,\ldots,T} g_\gamma(\xi^\pi_{x_0}, t). \tag{7.3}$$

For all $\gamma \in (0, 1)$, the super-zero level set of $V_\gamma(x, T)$, defined as $\mathcal{V}_\gamma := \{x : V_\gamma(x, T) > 0\}$, is equal to the RA set.

We leverage learning-based reachability methods (e.g., [73, 99]), which use deep reinforcement learning (RL) to learn a deterministic control policy π and value function V_γ. In the rest of this chapter, the learned policy is denoted by π_{lp}.

Scenario Optimization Theory

Scenario optimization is a data-driven framework used to provide probabilistic guarantees for optimization problems under uncertainty Δ:

$$\begin{aligned} \min_{z \in \mathcal{Z}} \quad & c^\top z \\ \text{s.t.} \quad & \mathbb{P}_\Delta\{h(z, \Delta) > 0\} \leq \epsilon \end{aligned} \tag{7.4}$$

where $z \in \mathcal{Z} \subseteq \mathbb{R}^d$ is the decision variable, h is convex, and $\epsilon \in (0, 1)$ is the maximum allowable risk. To solve (7.4) without assuming a specific distribution for Δ, we employ the *scenario approach*, replacing the chance constraint with N deterministic constraints sampled from Δ:

$$\begin{aligned} z_N^* = \arg\min_{z \in \mathcal{Z}} \quad & c^\top z \\ \text{s.t.} \quad & \bigwedge_{i=1}^{N} h(z, \delta^{(i)}) \leq 0 \end{aligned} \tag{7.5}$$

where each $\delta^{(i)}$ is an i.i.d. realization of Δ. For user-specified risk level ϵ and confidence parameter β, the required number of deterministic constraints N to achieve a probabilistic guarantee is given in Theorem 1.

Theorem 1 (Sample Complexity [26, 43])**.** *Let ϵ be the risk level and $\beta \in (0, 1)$ be the confidence parameter. Denote by $\mathbb{P}_S$ the probability measure associated with the number of scenarios $N \geq \frac{2}{\epsilon}\left(\ln \frac{1}{\beta} + d\right)$, taken from Δ and by $\mathbb{P}_\Delta$ the probability measure associated with Δ. Then z_N^* satisfies:*

$$\mathbb{P}_S(\mathbb{P}_\Delta(h(z_N^*, \delta) > 0) \leq \epsilon) \geq 1 - \beta. \tag{7.6}$$

Inequality (7.6) states that with confidence at least $1 - \beta$, an unseen $\delta \in \Delta$ violates h with probability at most ϵ. In practice, β is set small (e.g. 0.001) at low sample cost due to the log-inverse dependence, while ϵ trades off risk tolerance against sample complexity. The advantage of the scenario approach is therefore this established closed-form relationship between sample budget and guarantee: given ϵ and β, Theorem 1 fixes N a priori, providing structure to sampling effort.

7.3 Hybrid Reachability Verification Method

In this section, we describe our hierarchical verification framework. We present a scenario optimization-based technique to bound the deviation of trajectories. Then, we detail the construction of our global probabilistic certificate, and then introduce the online local refinement module used to expand this certificate near its boundary. Finally, the last part of this section integrates these components into our hierarchical condition-based switching architecture.

Probabilistic Dynamics Bounding

To construct a global probabilistically certified under-approximation for the RA set, we must first characterize the sensitivity of the system dynamics under the learned policy π_{lp}. The deterministic verification technique in [99] leverages the Lipschitz constant of the dynamics. Here, we alleviate conservatism by instead estimating a trajectory deviation quantity Δx_t^* using samples.

Let $\bar{\mathcal{X}} \subseteq \mathcal{X}$ denote the nominal state sampling domain. We collect N i.i.d. scenario pairs $\{(\bar{x}_0^{(i)}, x_0^{(i)})\}_{i=1}^N$ sampled uniformly, where each nominal state $\bar{x}_0^{(i)}$ is sampled uniformly from $\bar{\mathcal{X}}$, and each perturbed state $x_0^{(i)}$ is sampled from an ρ-ball $\mathcal{B}_\rho(\bar{x}_0^{(i)})$. We denote the resulting pair space by $\mathcal{X}_{pair}$. For each pair, we generate a nominal control sequence $\mathbf{U}^{(i)} = \{u_0^{(i)}, \ldots, u_{T-1}^{(i)}\}$ by applying π_{lp} to $\bar{x}_0^{(i)}$, yielding the nominal trajectory $\bar{\xi}^{(i)} := \{\bar{x}_t\}_{t=0}^T$. For computational simplicity, we approximate the control for $x_0^{(i)}$ using the same sequence $\mathbf{U}^{(i)}$, yielding the perturbed trajectory $\xi^{(i)} := \{x_t\}_{t=0}^T$.

The global sensitivity bound Δx_t^* is obtained by solving:

$$\begin{aligned} \min_{\Delta x_t \in \mathbb{R}} \quad & \Delta x_t \\ & \text{s.t. } \mathbb{P}_{\mathcal{X}_{pair}}\left(\|x_t - \bar{x}_t\|_2 > \Delta x_t\right) \leq \epsilon_{global}. \end{aligned} \tag{7.7}$$

We leverage scenario optimization theory to obtain the following scenario program:

$$\begin{aligned} \Delta x_t^* = \min_{\Delta x_t \in \mathbb{R}} \quad & \Delta x_t \\ \text{s.t.} \quad & \bigwedge_{i=1}^{N} \left(\|x_t^{(i)} - \bar{x}_t^{(i)}\|_2 \leq \Delta x_t\right) \end{aligned} \tag{7.8}$$

By Theorem 1, the trajectory deviation remains bounded by Δx_t^* with probability $1 - \epsilon_{global}$ and confidence $1 - \beta_{global}$:

$$\mathbb{P}_{S_1}\left(\mathbb{P}_{\mathcal{X}_{pair}}\left(\|x_t - \bar{x}_t\|_2 > \Delta x_t^*\right) \leq \epsilon_{global}\right) \geq 1 - \beta_{global} \tag{7.9}$$

where $\mathbb{P}_{\mathcal{S}_1}$ denotes the probability measure over the sampled scenario dataset used to construct the scenario program, while $\mathbb{P}_{\mathcal{X}_{pair}}$ denotes the probability measure associated with the underlying pair sampling distribution over nominal and perturbed initial conditions.

Coarse Global Verification

We leverage the trajectory deviation bound Δx_t^* derived in the previous section to obtain a probabilistically certified lower bound of the value function, denoted as $\check{V}_\gamma$, under learned policy π_{lp}, as illustrated in Theorem 2.

Theorem 2 (Global Probabilistic Safety)**.** *Let $x_0 \in \mathcal{B}_\rho(\bar{x}_0)$. Suppose that for each $t \in \{0, 1, \ldots, T\}$, Δx_t^* is as calculated above. Let $\check{r}_t := r(\bar{x}_t) - L_r \Delta x_t^*$ and $\check{c}_t := c(\bar{x}_t) - L_c \Delta x_t^*$. Define a certificate function $\check{V}_\gamma(\bar{x}_0, T)$,*

$$\check{V}_\gamma(\bar{x}_0, T) := \max_{t=0,\ldots,T} \min\{\gamma^t \check{r}_t, \min_{\tau=0,\ldots,t} \gamma^\tau \check{c}_\tau\}. \tag{7.10}$$

Then, the following probabilistic guarantee holds:

$$\mathbb{P}_{S_1}\Big(\mathbb{P}_{\mathcal{X}_{pair}}\big(\{V_\gamma(x_0, T) < 0\} \cap \{\check{V}_\gamma(\bar{x}_0, T) \geq 0\}\big) \leq \epsilon_{global}\Big) \geq 1 - \beta_{global}.$$

Proof. Let $t \in \{1, \ldots, T\}$. Similar to the analysis in Section VI of [99], we define the set of states originating from the ball $\mathcal{B}_\rho(\bar{x}_0)$ under the nominal controls $\{\pi_{lp}(\bar{x}_\tau)\}_{\tau=0}^{t-1}$ as $\mathcal{X}_t := \{x_t : x_{\tau+1} = f(x_\tau, \pi_{lp}(\bar{x}_\tau)), \tau \leq t-1, x_0 \in \mathcal{B}_\rho(\bar{x}_0)\}$. Observe that the set $\bar{\mathcal{X}}_{t,\bar{x}_0} := \{x_t : \|x_t - \bar{x}_t\|_2 \leq \Delta x_t^*\}$ is a convex outer approximation of $\mathcal{X}_t$. By Lipschitz continuity of $r(x)$, we have that with confidence $1 - \beta_{global}$,

$$\forall x_t \in \bar{\mathcal{X}}_{t,\bar{x}_0},\ \mathbb{P}_{\mathcal{X}_{pair}}(|r(x_t) - r(\bar{x}_t)| \leq L_r \Delta x_t^*) \geq 1 - \epsilon_{global}$$

which yields a lower bound of $r(x_t)$, for all $x_t \in \bar{\mathcal{X}}_{t,\bar{x}_0}$:

$$\mathbb{P}_{\mathcal{X}_{pair}}(\check{r}_t \leq r(x_t)) \geq 1 - \epsilon_{global}.$$

Similarly, define a lower bound of $c(x_t)$, for all $x_t \in \bar{\mathcal{X}}_{t,\bar{x}_0}$:

$$\mathbb{P}_{\mathcal{X}_{pair}}(\check{c}_t \leq c(x_t)) \geq 1 - \epsilon_{global}.$$

As shown in Eq. (7.10), we can use $\check{r}_t$ and $\check{c}_t$ to construct a probabilistic lower bound function $\check{V}_\gamma(\bar{x}_0, T)$ for $V_\gamma(x_0, T)$, for all $x_0 \in \mathcal{B}_\rho(\bar{x}_0)$. In other words, $\check{V}_\gamma(\bar{x}_0, T)$ serves as a lower bound of $V_\gamma(x_0, T)$ with probability $1 - \epsilon_{global}$ over the support $\mathcal{X}_{pair}$. Denote by X and Y the events $\{V_\gamma(x_0) < \check{V}_\gamma(\bar{x}_0, T)\}$ and $\{\check{V}_\gamma(\bar{x}_0, T) \geq 0\}$, respectively. Thus, we have with confidence at least $1 - \beta_{global}$, $\mathbb{P}_{\mathcal{X}_{pair}}(X) \leq \epsilon_{global}$. By the law of total probability, we have, with confidence at least $1 - \beta_{global}$,

$$\mathbb{P}_{\mathcal{X}_{pair}}(X \cap Y) + \mathbb{P}_{\mathcal{X}_{pair}}(X \cap Y^c) \leq \epsilon_{global},$$

and so $\mathbb{P}_{\mathcal{X}_{pair}}(X \cap Y) \leq \epsilon_{global}$. Since $\{V_\gamma(x_0, T) < 0\} \cap \{\check{V}_\gamma(\bar{x}_0, T) \geq 0\}$ implies $\{V_\gamma(x_0) < \check{V}_\gamma(\bar{x}_0, T)\} \cap \{\check{V}_\gamma(\bar{x}_0, T) \geq 0\}$ always, then with confidence at least $1 - \beta_{global}$,

$$\mathbb{P}_{\mathcal{X}_{pair}}(\{V_\gamma(x_0, T) < 0\} \cap \{\check{V}_\gamma(\bar{x}_0, T) \geq 0\}) \leq \epsilon_{global}. \qquad \square$$

Theorem 2 states that, under the sampling distribution $\mathcal{X}_{pair}$, the probability that the lower-bound certificate incorrectly classifies an unsafe state as safe is upper bounded by ϵ_{global} with confidence $1 - \beta_{global}$. We define the globally certified region $\mathcal{V}_{global}$, which acts as a conservative under-approximation of the true RA set, as:

$$\mathcal{V}_{global} := \{x \in \bar{\mathcal{X}} \mid \check{V}_\gamma(x, T) \geq 0\}. \tag{7.11}$$

For the global certificate, Δx_t^* is computed offline and $\mathcal{V}_{\text{global}}$ is evaluated online via a grid of resolution ρ.

Local Growth Iterative Refinement

Since $\mathcal{V}_{global}$ is an RA under-approximation, safe states may lie outside of it. We therefore introduce an online local refinement module, inspired by [103], that anchors a candidate ball $\mathcal{B}_r(\check{x}_t)$ to the nearest boundary point $\check{x}_t \in \partial\mathcal{V}_{global}$ to the current state and determines the maximum certifiable radius r^* via a sample-and-shrink scenario optimization:

$$\begin{aligned} r^* = \max_{r\in\mathbb{R}} \quad & r \\ \text{s.t.} \quad & \bigwedge_{j=1}^{N} \max_{t\in[0,T]} g_\gamma(\xi^{\pi_{lp}}_{x_0^{(j)}}, t) > 0, \quad \forall x_0^{(j)} \in \mathcal{B}_r(\check{x}_t). \end{aligned} \tag{7.12}$$

As shown in Algorithm 1, the refinement centers an initial ball of radius r on the nearest boundary point $\check{x}_t \in \partial\mathcal{V}_{global}$ (lines 1-2). For M iterations, N initial conditions within the region are sampled within the ball and simulated under the π_{lp} (lines 3-4). If a violation of (7.2) is found, the radius shrinks to exclude it (lines 5-6), and the process is repeated. If no violations are found, the ball $\mathcal{V}_{local} := \mathcal{B}_{r^*}(\check{x}_t)$ is returned (lines 7-10). By Theorem 1, the local certificate is:

$$\mathbb{P}_{S_2}\left(\mathbb{P}_{\mathcal{V}_{local}}\big(\max_{t\in[0,T]} g_\gamma(\xi^{\pi_{lp}}_{x_0^{(j)}}, t) \le 0\big) \le \epsilon_{local}\right) \ge 1 - \beta_{local} \tag{7.13}$$

where $\mathbb{P}_{S_2}$ denotes the probability measure associated with the samples, and the guarantee is with respect to the sampling distribution stating that the probability of failing to safely reach the target set is bounded by ϵ_{local} with high confidence.

Remark (Construction vs. Verification): If Algorithm 1 breaks at iteration $\tau \le M$, we regard iterations $0, \ldots, \tau - 1$ as a *construction* phase that adaptively proposes a candidate radius, and iteration τ as a *verification* phase: a single scenario optimization instance on a fresh i.i.d. batch drawn against the fixed r^*. Certifying the full construction-then-verification pipeline is left to future work.

Hierarchical Verification and Control Framework

Our verification framework features a condition-based hybrid switching logic that prioritizes the highest available tier of probabilistic certification. When the state lies within a certified region, π_{lp} is executed; otherwise, control authority is delegated to a model-based fallback, Model Predictive Path Integral (MPPI) [159] warm-started with π_{lp}. Since optimal RA policies may not guarantee target set invariance [34], MPPI is also used once $x_t \in \mathcal{T}$. The hierarchical architecture is illustrated in Fig. 7.1 and Algorithm 2, detailing four modes:

1. **Tier 0: Target Maintenance Mode:** If $x_t \in \mathcal{T}$, MPPI maintains the system within $\mathcal{T}$ (reach-avoid-and-stay).

Algorithm 1: IterativeGrowth

Initialize: x_t, π_{lp}, $\mathcal{V}_{global}$, N, r_{max}, M, T
Output: r_{safe}, $\mathcal{V}_{local}$
Initialize: $r^* \leftarrow r_{max}$
Find closest boundary point $\check{x} \in \partial\mathcal{V}_{global}$ to x_t
for iter=$0, 1, \ldots, M-1$ **do**
    $\mathcal{S}_i \leftarrow$ Sample N states IID from $\mathcal{B}_r(\check{x}_t)$, roll out trajectories $\xi_x^{\pi_{lp}}$.
    if $\exists x \in \mathcal{S}_i$ s.t. $\max_{t\in[0,T]} g_\gamma(\xi_x^{\pi_{lp}}, t) \leq 0$ **then**
        $r^* \leftarrow \min_x \|x - \check{x}_t\|_2$ **s.t.** $\max_{t\in[0,T]} g_\gamma(\xi_x^{\pi_{lp}}, t) \leq 0$
    else
        return $\mathcal{V}_{local} := \mathcal{B}_{r^*}(\check{x}_t)$
    end if
end for

2. **Tier 1: Global Verification Mode:** If $x_t \in \mathcal{V}_{\text{global}}$, the system is within the globally, scenario-certified region, and $\pi_{lp}(x_t)$ is executed.

3. **Tier 2: Local Refinement Mode:** If $x_t \notin \mathcal{V}_{global}$ and $x_t \in \mathcal{V}_{local}$, the online local certificate confirms π_{lp} is reliable beyond the global boundary and is executed.

4. **Tier 3: Safety Recovery Mode:** If $x_t \notin \mathcal{V}_{global}$ and $x_t \notin \mathcal{V}_{local}$, MPPI warm-started by π_{lp} minimizes distance to the nearest verified set boundary, steering the system into a certified region.

Parameter Selection and Trade-offs

The framework is parameterized by N, M, T, and ρ. Sample size N follows Theorem 1: smaller ϵ, β increase N, yielding a more conservative $\mathcal{V}_{\text{global}}$ and $\mathcal{V}_{\text{local}}$. The iteration cap M bounds online sample-and-shrink, giving Tier 2 complexity $\mathcal{O}(MNT)$. The horizon T trades off error against rollout cost: larger T reduces finite-time approximation error but increases computation. A smaller grid resolution ρ of the global certificate reduces conservatism but increases computation and should be chosen subject to real-time budget constraints.

7.4 Theoretical Reach-avoid Guarantees

Tier 1 & Tier 2: Probabilistic RA Set Verification

Given the hierarchical nature of the framework, we consider sequential probabilistic guarantees leveraged as needed.

Proposition 1 (Sequential Probabilistic Guarantees)**.** *Let $\mathcal{P}_{global}$ and $\mathcal{P}_{local}$ denote the global and local scenario sampling distributions, respectively.*

Algorithm 2: Hierarchical Verification Framework

Initialize: x_t, $\mathcal{V}_{global}$, π_{lp}
Output: u_t

$\mathcal{T} \leftarrow \textsc{UpdateSet}(x_t)$
if $x_t \in \mathcal{T}$ **then**
    return $\textsc{MPPI_Fast}(x_t)$ *// Tier 0*
else if $x_t \in \mathcal{V}_{global}$ **then**
    return $\pi_{lp}(x_t)$ *// Tier 1*
else
    $\mathcal{V}_{local} \leftarrow \textsc{IterativeGrowth}(x_t, \pi_{lp}, \mathcal{V}_{global})$
    if $x_t \in \mathcal{V}_{local}$ **then**
        return $\pi_{lp}(x_t)$ *// Tier 2*
    else
        return $\textsc{MPPI_Rec}(\text{Warmstart} = \pi_{lp})$ *// Tier 3*
    end if
end if

Global certificate: *The global verified region $\mathcal{V}_{global}$ satisfies the following probabilistic certification:*

$$\mathbb{P}_{S_1}\Big(\mathbb{P}_{\mathcal{P}_{global}}\big(\{V_\gamma(x_0) < 0\} \cap \{\check{V}_\gamma(\bar{x}_0, T) \geq 0\}\big) \leq \epsilon_{global}\Big) \geq 1 - \beta_{global}.$$

Local certificate: *For states sampled from the local refinement distribution,*

$$\mathbb{P}_{S_2}\left(\mathbb{P}_{\mathcal{P}_{local}}\big(\max_{t\in[0,T]} g_\gamma(\xi^{\pi_{lp}}_{x_0^{(j)}}, t) \leq 0\big) \leq \epsilon_{local}\right) \geq 1 - \beta_{local}.$$

Proof. This follows from Theorems 1 and 2. □

Remark (Interpretation): ϵ_{global} and ϵ_{local} bound violation probabilities under $\mathcal{P}_{global}$ and $\mathcal{P}_{local}$, not the closed-loop runtime distribution. These certificates quantify the reliability of π_{lp} under these assumptions and are used to probabilistically guide decision-making. We demonstrate the usefulness of certificate-guided decision-making during runtime in our experiments in Section 7.5.

Tier 3: Finite-time Recovery

If $x_t \notin (\mathcal{V}_{global} \cup \mathcal{V}_{local})$, no safety guarantees can be made and recovery via reachability-warm-started MPPI is invoked. Let $\mathcal{V}_{recover}$ denote the forward reachable set of the ego agent at x_t. Then, recovery is feasible when $\mathcal{V}_{recover}$ satisfies $\mathcal{V}_{recover} \cap (\mathcal{V}_{global} \cup \mathcal{V}_{local}) \neq \varnothing$.

7.5 Experiments and Results

We evaluate our framework on a high-dimensional drone-racing task. Given the high dimensionality of the state, fine-grained, grid-based verification baselines are intractable. Therefore, comparison against model-based and learning-based controllers is the standard benchmarking approach. A link to our code can be found at `https://github.com/ebonyelsmith/Hierarchical_Probabilistic_Verification`.

Drone Racing Case Study

We consider a drone racing example as defined in [99], a challenging high-dimensional testbed with dynamic interactions and safety-critical constraints, where the ego agent must safely overtake an opponent to pass through the gate first. We set $\beta_{global} = \beta_{local} = 0.001$ and $\epsilon_{global} = \epsilon_{local} = 0.1$ as the smallest values keeping N within our online sample budget, and so here $N = 158$.

Joint System Dynamics

We consider a 12D non-cooperative racing scenario involving an ego drone (e) and an opponent drone (o). The state of each drone $i \in \{e, o\}$ is defined by its 3D position and velocity: $x_t^i = [p_{x,t}^i, v_{x,t}^i, p_{y,t}^i, v_{y,t}^i, p_{z,t}^i, v_{z,t}^i] \in \mathbb{R}^6$. The joint system state is defined as the concatenation $x_t = [x_t^e, x_t^o] \in \mathcal{X} \subseteq \mathbb{R}^{12}$. The i-th drone is modeled by double-integrator dynamics, with control input $u_t^i = [a_{x,t}^i, a_{y,t}^i, a_{z,t}^i]$ satisfying $u_{t\infty}^i \leq \epsilon_u := 1\,\mathrm{m/s^2}$. We assume the opponent has bounded control authority and follows an LQR controller that drives it toward the gate aggressively, with high speed.

Target Set

The mission objective is to pass through a narrow gate centered at $[0, 0, 0]$ with width w and height h. However, for the purpose of reachability, the target set for the ego drone is defined as

$$\mathcal{T} = \left\{ \begin{array}{ll} x : p_y^e - p_y^o > 0, & v_y^e - v_y^o > 0, \\ |p_x^e| < 0.3, & |p_z^e| < 0.3 \end{array} \right\}. \tag{7.14}$$

This target set requires the ego drone to be ahead of and faster than the opponent while staying within a corridor for gate passage (radius 0.3 m).

Safety Constraints

To ensure safe flight, the ego drone should avoid the area affected by the downward airflow from the opponent drone, leading to the following nonconvex constraint, which is challenging to satisfy:

$$\left\| \begin{bmatrix} p_{x,t}^e - p_{x,t}^o \\ p_{y,t}^e - p_{y,t}^o \end{bmatrix} \right\|_2^2 > \Big(1 + \max(p_{z,t}^o - p_{z,t}^e, 0)\Big) \times 0.2, \tag{7.15}$$

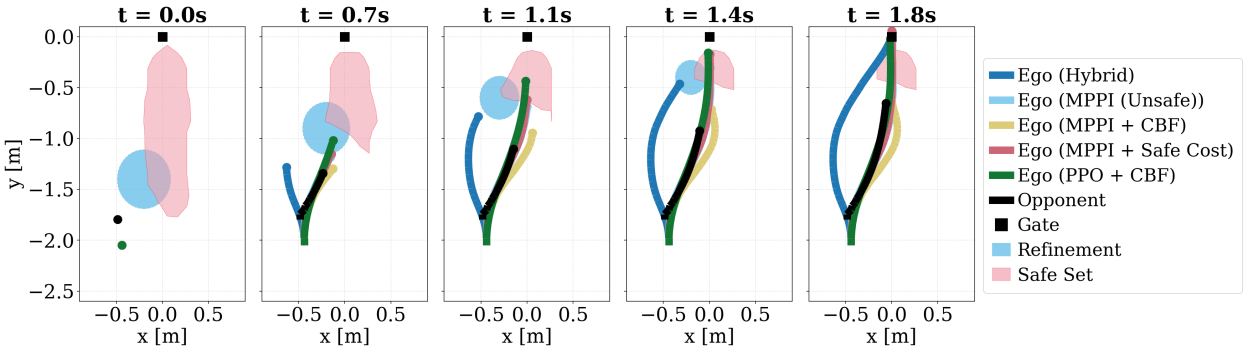


Figure 7.2: Evolution of a drone racing simulation against an opponent (black), comparing our method (Hybrid, dark blue) with various baselines. Baselines struggle to balance safety and goal-reaching, whereas our method achieves both, safely overtaking and reaching the gate. Moreover, we observe that the ego agent leverages safe recovery MPPI to try to enter the locally refined set.

where the collision avoidance region between the ego and opponent drone grows as their height difference increases. Moreover, to ensure the ego drone passes through the gate without colliding with its boundaries, we impose the following constraints as in [99]:

$$\pm p^e_{x,t} - p^e_{y,t} > -0.05, \quad \pm p^e_{z,t} - p^e_{y,t} > -0.05. \tag{7.16}$$

Hypothesis 1: The reachability-based guidance of our method leads to a higher success rate than soft- and hard-constrained model-based controllers or RL policies.

We empirically compare our framework to various safe baseline methods: MPPI with a control barrier function (CBF) safety filter, MPPI with soft safety constraints, and proximal policy optimization (PPO) with a CBF safety filter. We sample 500 initial conditions in a bounded region of the state space and roll out the trajectories. Success is quantified by the ego safely reaching the gate before the opponent, and our framework achieves a higher rate of success than its safe counterparts as shown in Figure 7.3(a). Qualitatively, baseline methods tend to prioritize goal-reaching at the expense of safety, whereas our method demonstrates more robust collision avoidance during overtaking (Fig. 7.2).

Hypothesis 2: The hierarchical control framework features a better combination of MPPI and learning.

Similar to Section 7.5, we conduct an empirical study comparing our framework to baseline methods: Hybrid Ablation which conducts switching but replaces the reachability policy with

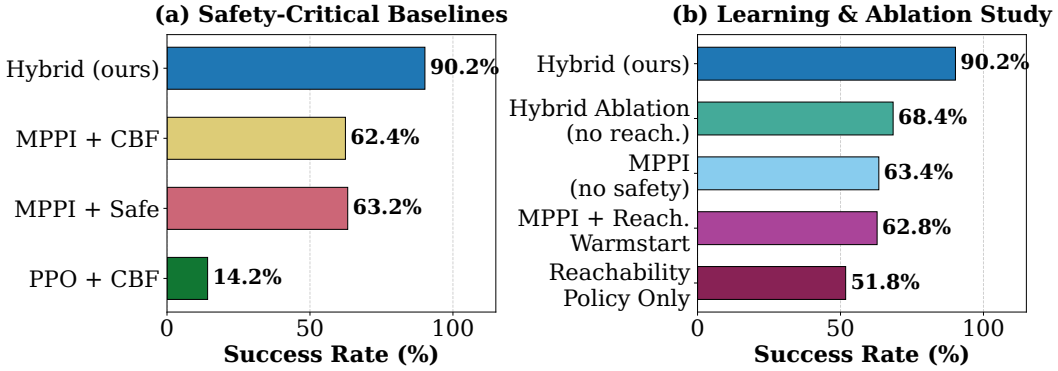


Figure 7.3: (a) Performance of our framework compared to control barrier function (CBF) and soft-constraint baselines. (b) Ablation study comparing the hybrid method to its learning-only and model-based variants. In both (a) and (b), the results for our method are shown for reference. *Our framework achieves higher success probability over 500 sampled initial conditions, effectively balancing collision avoidance and goal-reaching.*

MPPI, pure MPPI with no safety consideration, MPPI warm-started with the reachability policy, and the reachability policy only. We select these baselines to observe the interaction between MPPI, the reachability policy, and the switching mechanism. We sample 500 initial conditions in a bounded region of the state space and roll out the trajectories. The results in Figure 7.3(b) highlight the necessity of both the switching mechanism and the reachability policy as none of the other baselines achieved the same rate of success as our hierarchical framework. Notably, the reachability policy prioritizes reaching the target set but does not guarantee maintaining the system within it, a limitation inherent to reachability theory, as also observed in [34]. This explains its lower success rate when used alone.

7.6 Conclusions

In this work, we propose an efficient hierarchical probabilistic verification framework for reachability learning that unifies global, local, offline, and online scenario-based certification. By combining a coarse global certificate via scenario optimization with an online local convex refinement, the approach reduces conservatism while focusing computation on critical regions of the state space. A condition-based switching mechanism integrates the learned reachability policy with a model-based MPPI controller, enabling adaptive decision-making when certified regions are unavailable. The resulting framework provides scenario-based probabilistic certificates that quantify the reliability of the learned policy under the sampling distributions used in verification. These certificates are used to guide hierarchical control switching. Future work includes hardware validation and extension to more complex multi-agent settings and uncertain real-world environments.

Part III

Conclusion

Chapter 8

Discussion and Future Directions

This dissertation has discussed the design of assurance frameworks as divided into two parts: (1) runtime safety filtering for autonomous systems with machine learning components, and (2) statistical verification to obtain probabilistic guarantees of safety.

In the first part, I explored the use of uncertainty quantification from the machine learning literature for safety-critical control systems. Recognizing that uncertainty quantification for standalone modules can be ill-defined, I instead took a system-level perspective and argued that uncertainty should be considered using the language of design: design objectives, specifications, and vulnerabilities. I discussed a novel robustness metric for uncertainty-aware systems that can be used to compare designs and also make explicit the regions in which a system does or does not perform well. I also presented a new technique for generating a specification on a module's uncertainty measure given a system-level specification. Doing so contextualizes module uncertainty measures in terms of the downstream decision-making and system objectives. Thus, the methodology also presents a way to quantify how well a decision-making algorithm uses an uncertainty measure.

In the second part, I presented three different studies on how statistical tools such as conformal prediction and scenario optimization can be used in control contexts to probabilistically certify control and adaptation. First, I introduced a safe control pipeline that produces probabilistically safe actions in the presence of other agents even when using a learned predictor module that is subject to error. The technique uses a multi-step procedure to produce an uncertainty representation, calibrate it with conformal prediction, and then use that calibrated representation downstream. Second, I discussed an active learning algorithm that can be used to learn or adapt a model to new conditions while simultaneously producing a generalization bound. Finally, if a safe controller was synthesized according to a learning procedure, the estimate of how safe it will be during deployment may be prone to errors. I discussed a hierarchical framework to certify regions of the state space to guide the controller at deployment time. Since the global probabilistic guarantee may be overly conservative, a local certificate is applied where it is most likely to be relevant to the controller and decrease conservatism.

8.1 Advances and Limitations

The works discussed in this dissertation achieve the objective of producing a better understanding of how machine learning and statistics tools such as uncertainty quantification and probabilistic bounds can be used to ensure the safety of control systems. Part I bridges gaps between autonomous system theory and practice by proposing new, principled analytical tools for real-world system designs. Part II proposes novel data-driven frameworks for safety that, by being more adaptive and computationally lightweight than many traditional methods, show practical promise for the complex and opaque machine learning-based systems of today.

However, some of the proposed methods rely only on data for analysis or assurance. This is a good approach when systems or components are black-box, but a more complete picture of safety might involve both data and some knowledge of module internals. Therefore, data should be only one of multiple ingredients in an assurance framework. Also, many of the tools introduced here are not applicable for closed-loop decision-making or multi-agent scenarios. Although the methods in Part II address probabilistic guarantees in dynamic environments or when the system is learning, none of the methods in Parts I or II address cases in which a system plans or updates itself to reduce uncertainty, or the extreme nonstationarity that can arise when other agents in the environment react or adapt to the ego agent.

8.2 Directions for Future Work

Assurance Frameworks for the Open World. The "open world" in the context of autonomous systems refers to the unstructured and complex environments that humans operate in on a daily basis, involving interactions with others, unexpected situations, and challenges that require us to learn and adapt. However, producing assurances requires making assumptions of some kind on what the world looks like. In deterministic safety tools such as classic Hamilton-Jacobi reachability, for example, assumptions are made on system dynamics and the structure of the environment, and safety is a well-defined and fixed concept. Meanwhile, as mentioned in the Introduction, probabilistic methods must make assumptions on the distribution that the test data points come from, and the nature of the guarantee is also highly sensitive to the system objectives and definition of safety. Also as mentioned in the Introduction, while data-driven assurance frameworks help to approximate the world in which a system will operate, safety in the face of the truly unpredictable remains an unsolved problem.

Dealing with the open world is clearly a very broad research direction. One concrete idea relates to the need to create *multifaceted* assurance frameworks that not only are based on data-driven probabilistic guarantees, but also include, perhaps, *semantics* relating to where and why a system may fail. Probabilistic guarantees essentially summarize the data that was used to create them, but it is simultaneously important to provide more fine-grained, per-data-point safety understanding. It may be necessary to create new frameworks that

unite previously disparate methods and tools, such as statistical verification, mechanistic interpretability, dataset visualization, simulation, and formal proofs. Doing so may help to alleviate the assumptions made by each individual method; none of these alone can give a complete picture of the world.

Recent years have also seen scaling up to be a way to address the open-world issue — training larger models with more data so that they can make accurate predictions for more situations. Another, perhaps complementary, approach is *adaptation* to the unknown. This is especially important as we ground assurance frameworks by bringing humans into the picture: assurances, however they are produced, ultimately have to be satisfying to end users and stakeholders. *Safety* is therefore a concept subservient to *trust* between human and autonomous system. Trust requires adaptation, not only to new environments and types of tasks, but also to new preferences, definitions of safety, and degrees of transparency as they depend on the user. Future work should focus not only on individual assurance frameworks or specific metrics that a system should optimize, but on effectively building trust over time, much in the same way one would control a dynamical system. For this, we need a better understanding of how different pieces of an assurance framework interact with each other. For example, if one changes the system objectives or constraints slightly, what is the change to the assurance statement? This dissertation has made arguments for considering uncertainty quantification as a system design problem. Assurance frameworks in general should be treated similarly.

Safety and Foundation Models. With the steadily growing adoption of foundation models for autonomous systems, new interesting problems arise for assurance frameworks. These can be divided into two categories: (1) augmenting safety and trust using foundation models, and (2) analyzing the safety of the foundation models themselves.

With their internet-scale knowledge and ability to perform "common-sense reasoning", foundation models have the potential to play an instrumental role in certifying other models or systems. For example, they can be used in generative applications to produce test cases or entire simulations, reducing the cost of acquiring data to produce assurances. They can be used to provide explanations of other models' behavior in the context of the environmental scenario, or comprise more advanced runtime safety filters that reason about whether anything in the environment may cause the system to fail and what to do in response. These large models can also extend the capabilities of existing safety tools such as Hamilton-Jacobi reachability by interfacing with more complex system dynamics, constraints, or objectives. Indeed, several recent works have already started studying these ideas [70, 121, 141, 157]. Future work can continue exploring these directions; one idea for example is to examine how to train foundation models to better understand safety concepts. How can foundation models better capture different priority levels or definitions of safety, especially when these depend on the downstream application? Can they aid in elucidating how safety should be enforced, for example as hard constraints, soft constraints, or probabilistically? Coming to the concept of trust, how can foundation models aid in facilitating communication between humans and autonomous systems, for example by using language to express the right kinds

of uncertainty or accurately capture intents and action consequences? As the usage of foundation models for autonomy grows, this list of questions to explore will undoubtedly grow as well.

What is interesting about using foundation models to augment safety, however, is that they are themselves machine learning models without guarantees on their outputs. Therefore, the second category of future work relates to verifying foundation models. One interesting observation is that a *bilevel* verification problem arises when a foundation model that is being used to verify another model needs to itself be verified. What is the nature of such a problem and how can it be solved? Which tools will need to be involved? This dissertation has explored uncertainty quantification and statistical verification as ways to produce assurances for systems involving machine learning. How can these tools be applied or, possibly, expanded when solving both single-level and bilevel verification problems? For one, many of the uncertainty quantification techniques mentioned in this dissertation may be too computationally intensive to apply to foundation models or be especially difficult to interpret. Some foundation models additionally have language outputs, for which useful and powerful uncertainty representations may be very challenging to create. Meanwhile, performing statistical verification techniques for foundation models may require a prohibitive amount of samples since the models have vast, high-dimensional operational domains. These questions all pose important challenges for foundation model-enabled systems.

As learning-enabled autonomous systems become more powerful and more intertwined with our daily lives, there will be many situations in which humans will need to influence, understand, and be assured of the systems' operation. It will be increasingly important to maintain a *spectrum* of assurance frameworks so that, depending on the application, one may choose frameworks ranging from those with few assurances to those with a large number of constraints that are guaranteed to be satisfied at all times. This requires a deep understanding of the *design* of assurance frameworks so that we may take principled and thoughtful steps into the new era.

Bibliography

[1] Matthew Abate and Samuel Coogan. "Robustly forward invariant sets for mixed-monotone systems". In: *IEEE Transactions on Automatic Control* 67.9 (2022), pp. 4947–4954.

[2] Moloud Abdar et al. "A Survey of Uncertainty in Deep Neural Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1 (2023), pp. 3–27.

[3] Gul Agha and Karl Palmskog. "A Survey of Statistical Model Checking". In: *ACM Transactions on Modeling and Computer Simulation* 28.1 (2018). DOI: 10.1145/3158668.

[4] Sankalp Agrawal et al. "AnySafe: Adapting Latent Safety Filters at Runtime via Safety Constraint Parameterization in the Latent Space". In: *CoRR* abs/2509.19555 (2025). DOI: 10.48550/arXiv.2509.19555. URL: https://arxiv.org/abs/2509.19555.

[5] Prithvi Akella and Aaron D Ames. "Disturbance bounds for signal temporal logic task satisfaction: A dynamics perspective". In: *IEEE Control Systems Letters* 6 (2021), pp. 2018–2023.

[6] Matthias Althoff and Bruce H Krogh. "Zonotope bundles for the efficient computation of reachable sets". In: *2011 50th IEEE conference on decision and control and European control conference*. IEEE. 2011, pp. 6814–6821.

[7] Rajeev Alur. *Principles of Cyber-Physical Systems*. Cambridge, MA: MIT Press, 2015.

[8] Siddharth Ancha, Philip R. Osteen, and Nicholas Roy. "Deep Evidential Uncertainty Estimation for Semantic Segmentation under Out-Of-Distribution Obstacles". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 6943–6951. DOI: 10.1109/ICRA57147.2024.10611342.

[9] Anastasios N. Angelopoulos and Stephen Bates. "Conformal Prediction: A Gentle Introduction". In: *Foundations and Trends in Machine Learning* 16.4 (2023), pp. 494–591. DOI: 10.1561/2200000101.

[10] Shervin Ardeshir and Navid Azizan. "Uncertainty in Contrastive Learning: On the Predictability of Downstream Performance". In: *CoRR* abs/2207.09336 (2022). arXiv: 2207.09336. URL: https://doi.org/10.48550/arXiv.2207.09336.

[11] Andrea Bajcsy et al. "A scalable framework for real-time multi-robot, multi-human collision avoidance". In: *2019 International Conference on Robotics and Automation*. IEEE. 2019, pp. 936–943.

[12] Andrea Bajcsy et al. "An Efficient Reachability-Based Framework for Provably Safe Autonomous Navigation in Unknown Environments". In: *Proceedings of the IEEE Conference on Decision and Control (CDC)*. 2019, pp. 1758–1765. DOI: 10.1109/CDC40024.2019.9030133.

[13] Somil Bansal and Claire J. Tomlin. "DeepReach: A Deep Learning Approach to High-Dimensional Reachability". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1817–1824. DOI: 10.1109/ICRA48506.2021.9561949.

[14] Somil Bansal et al. "Combining Optimal Control and Learning for Visual Navigation in Novel Environments". In: *Proceedings of the Conference on Robot Learning (CoRL)*. Vol. 100. Proceedings of Machine Learning Research. PMLR, 2020, pp. 420–429. URL: https://proceedings.mlr.press/v100/bansal20a.html.

[15] Somil Bansal et al. "Hamilton-jacobi reachability: A brief overview and recent advances". In: *2017 IEEE 56th Annual Conference on Decision and Control*. IEEE. 2017, pp. 2242–2253.

[16] Rina Foygel Barber et al. "Conformal Prediction Beyond Exchangeability". In: *The Annals of Statistics* 51.2 (2023), pp. 816–845. DOI: 10.1214/23-AOS2276.

[17] Michèle Basseville. "Detecting changes in signals and systems—a survey". In: *Automatica* 24.3 (May 1988), pp. 309–326. ISSN: 0005-1098. DOI: 10.1016/0005-1098(88)90073-8. URL: https://doi.org/10.1016/0005-1098(88)90073-8.

[18] Keni Bernardin and Rainer Stiefelhagen. "Evaluating multiple object tracking performance: the CLEAR MOT metrics". In: *EURASIP J. Image Video Process.* 2008 (Jan. 2008). ISSN: 1687-5176. DOI: 10.1155/2008/246309. URL: https://doi.org/10.1155/2008/246309.

[19] Mogens Blanke et al. *Diagnosis and Fault-Tolerant Control*. 3rd ed. Berlin, Heidelberg: Springer, 2016. DOI: 10.1007/978-3-662-47943-8.

[20] Laurence BonJour. *The Structure of Empirical Knowledge*. Cambridge, MA, USA: Harvard University Press, 1985.

[21] Javier Borquez, Kensuke Nakamura, and Somil Bansal. "Parameter-Conditioned Reachable Sets for Updating Safety Assurances Online". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2023. URL: https://arxiv.org/abs/2209.14716.

[22] Dimitrios Boursinos and Xenofon Koutsoukos. "Assurance monitoring of learning-enabled cyber-physical systems using inductive conformal prediction based on distance learning". In: *AI EDAM* 35.2 (2021), pp. 251–264.

[23] Sébastien Bubeck and Nicolò Cesa-Bianchi. *Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems*. Vol. 5. Foundations and Trends in Machine Learning, 2012, pp. 1–122. DOI: 10.1561/2200000024. URL: https://www.nowpublishers.com/article/Details/MAL-024.

[24] Holger Caesar et al. "nuScenes: A multimodal dataset for autonomous driving". In: *Conference on Computer Vision and Pattern Recognition*. 2020.

[25] Xiaoyi Cai et al. "EVORA: Deep Evidential Traversability Learning for Risk-Aware Off-Road Autonomy". In: *IEEE Transactions on Robotics* 40 (2024), pp. 3756–3777. DOI: 10.1109/TRO.2024.3431828.

[26] Marco C Campi, Simone Garatti, and Maria Prandini. "The scenario approach for systems and control design". In: *Annual Reviews in Control* 33.2 (2009), pp. 149–157.

[27] Marco C. Campi and Simone Garatti. "Compression, Generalization and Learning". In: *Journal of Machine Learning Research* 24.339 (2023), pp. 1–74.

[28] Marco C. Campi and Simone Garatti. *Introduction to the Scenario Approach*. MOS-SIAM Series on Optimization. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2018. DOI: 10.1137/1.9781611975444.

[29] Michael E Cao, Matthieu Bloch, and Samuel Coogan. "Estimating high probability reachable sets using gaussian processes". In: *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE. 2021, pp. 3881–3886.

[30] Devendra Singh Chaplot et al. "Learning to Explore Using Active Neural SLAM". In: *International Conference on Learning Representations (ICLR)*. 2020. URL: https://openreview.net/forum?id=HklXn1BKDH.

[31] Bertrand Charpentier et al. "Disentangling Epistemic and Aleatoric Uncertainty in Reinforcement Learning". In: *CoRR*. 2022.

[32] Li Chen et al. "End-to-End Autonomous Driving: Challenges and Frontiers". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.12 (2024), pp. 10117–10137. DOI: 10.1109/TPAMI.2024.3435937.

[33] Yuxiao Chen et al. "Reactive motion planning with probabilistic safety guarantees". In: *Conference on Robot Learning*. PMLR. 2021, pp. 1958–1970.

[34] Gabriel Chenevert et al. "Solving reach-avoid-stay problems using deep deterministic policy gradients". In: *Accepted by IEEE international conference on robotics and automation (ICRA)* (2026).

[35] Chu Xin Cheng et al. "Practical Bayesian Algorithm Execution via Posterior Sampling". In: *Proceedings of the Thirty-Seventh Conference on Neural Information Processing Systems (NeurIPS 2024)*. Neural Information Processing Systems Foundation, Inc., 2024. DOI: 10.52202/079017-4296.

[36] Roderick M. Chisholm. *Theory of Knowledge*. 3rd. Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.

[37] Samuel Coogan. "Mixed monotonicity for reachability and safety in dynamical systems". In: *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE. 2020, pp. 5074–5085.

[38] Anthony L. Corso et al. "Risk-Driven Design of Perception Systems". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 9894–9906.

[39] Sampada Deglurkar et al. "Active Calibration of Reachable Sets Using Approximate Pick-to-Learn". In: *IEEE Control Systems Letters* (2026). DOI: 10.1109/LCSYS.2026.3710205.

[40] Sampada Deglurkar et al. "System-Level Analysis of Module Uncertainty Quantification in the Autonomy Pipeline". In: *2024 IEEE 63rd Conference on Decision and Control (CDC)*. IEEE, 2024, pp. 1103–1109. ISBN: 979-8-3503-1317-5. DOI: 10.1109/CDC56724.2024.10886366.

[41] Sampada Deglurkar et al. "System-Level Analysis of Module Uncertainty Quantification in the Autonomy Pipeline". In: *arXiv preprint arXiv:2410.12019v2* (2024). DOI: 10.48550/arXiv.2410.12019. arXiv: 2410.12019v2 [cs.RO]. URL: https://arxiv.org/abs/2410.12019v2.

[42] Ankush Desai et al. "SOTER: A Runtime Assurance Framework for Programming Safe Robotics Systems". In: *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. June 2019.

[43] Alex Devonport and Murat Arcak. "Estimating reachable sets with scenario optimization". In: *Learning for dynamics and control*. PMLR. 2020, pp. 75–84.

[44] Alex Devonport et al. "Data-driven reachability analysis with Christoffel functions". In: *2021 60th IEEE Conference on Decision and Control*. IEEE. 2021, pp. 5067–5072.

[45] Elizabeth Dietrich, Hanna Krasowski, and Murat Arcak. "Probably Approximately Correct (PAC) Guarantees for Data-Driven Reachability Analysis: A Theoretical and Empirical Comparison". In: *arXiv preprint arXiv:2604.02953* (2026). DOI: 10.48550/arXiv.2604.02953.

[46] Elizabeth Dietrich et al. "Data-driven reachability with scenario optimization and the holdout method". In: *IEEE 64th Conference on Decision and Control (CDC)*. IEEE. 2025, pp. 3925–3931.

[47] Liuhui Ding et al. "Capture Uncertainties in Deep Neural Networks for Safe Operation of Autonomous Driving Vehicles". In: *2021 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA/BDCloud/SocialCom/SustainCom)*. 2021, pp. 826–835. DOI: 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00118.

[48] Wenhao Ding et al. "RealGen: Retrieval Augmented Generation for Controllable Traffic Scenarios". In: *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXII*. Milan, Italy: Springer-Verlag, 2024, pp. 93–110. ISBN: 978-3-031-73032-0. DOI: 10.1007/978-3-031-73033-7_6. URL: https://doi.org/10.1007/978-3-031-73033-7_6.

[49] Anushri Dixit et al. "Adaptive Conformal Prediction for Motion Planning among Dynamic Agents". In: *Proceedings of the Learning for Dynamics and Control Conference (L4DC)*. Vol. 211. Proceedings of Machine Learning Research. PMLR, 2023, pp. 300–314. URL: https://proceedings.mlr.press/v211/dixit23a.html.

[50] Tommaso Dreossi et al. "VerifAI: A Toolkit for the Formal Design and Analysis of Artificial Intelligence-Based Systems". In: *31st International Conference on Computer Aided Verification (CAV)*. July 2019.

[51] Jr. Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. Cambridge, MA: MIT Press, 1999.

[52] Avner Engel. *Verification, Validation, and Testing of Engineered Systems*. Hoboken, NJ: John Wiley & Sons, 2010. ISBN: 9780470527511. DOI: 10.1002/9780470618851.

[53] Scott Ettinger et al. "Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset". In: *International Conference on Computer Vision*. IEEE. Oct. 2021, pp. 9710–9719.

[54] Alec Farid, Sushant Veer, and Anirudha Majumdar. "Task-Driven Out-of-Distribution Detection with Statistical Guarantees for Robot Learning". In: *Proceedings of the 5th Conference on Robot Learning*. Ed. by Aleksandra Faust, David Hsu, and Gerhard Neumann. Vol. 164. Proceedings of Machine Learning Research. PMLR, Aug. 2022, pp. 970–980. URL: https://proceedings.mlr.press/v164/farid22a.html.

[55] Sebastian Farquhar and Yarin Gal. "What 'Out-of-distribution' Is and Is Not". In: *ML Safety Workshop at NeurIPS* (Dec. 2022).

[56] Shai Feldman et al. "Achieving Risk Control in Online Learning Settings". In: *Transactions on Machine Learning Research* (2023).

[57] Jaime F Fisac et al. "Bridging hamilton-jacobi safety analysis and reinforcement learning". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 8550–8556.

[58] Jaime F Fisac et al. "Reach-avoid problems with time-varying dynamics, targets and constraints". In: *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*. 2015, pp. 11–20.

[59] Hamish Flynn et al. "PAC-Bayes Bounds for Bandit Problems: A Survey and Experimental Comparison". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.12 (2023), pp. 15308–15327. DOI: 10.1109/TPAMI.2023.3305381. URL: https://doi.org/10.1109/TPAMI.2023.3305381.

[60] P. Frazier. "A Tutorial on Bayesian Optimization". In: *ArXiv* abs/1807.02811 (2018). URL: https://api.semanticscholar.org/CorpusID:49656213.

[61] David Fridovich-Keil et al. "Confidence-aware motion prediction for real-time collision avoidance". In: *The International Journal of Robotics Research* 39.2-3 (2020), pp. 250–265.

[62] Yarin Gal. "Uncertainty in Deep Learning". PhD dissertation. University of Cambridge, 2016.

[63] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning". In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1050–1059.

[64] Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023. ISBN: 9781108348973. DOI: 10.1017/9781108348973.

[65] Isaac Gibbs and Emmanuel Candes. "Adaptive conformal inference under distribution shift". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 1660–1672.

[66] Isaac Gibbs and Emmanuel J. Candès. "Conformal Inference for Online Prediction with Arbitrary Distribution Shifts". In: *Journal of Machine Learning Research* 25.162 (2024), pp. 1–36.

[67] Alkis Gotovos et al. "Active learning for level set estimation". In: *IJCAI International Joint Conference on Artificial Intelligence* (Aug. 2013), pp. 1344–1350.

[68] Junru Gu, Chen Sun, and Hang Zhao. "Densetnt: End-to-end trajectory prediction from dense goal sets". In: *International Conference on Computer Vision*. IEEE. 2021, pp. 15303–15312.

[69] Joris Guerin et al. "Out-of-Distribution Detection Is Not All You Need". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 12. 2023, pp. 14829–14837. DOI: 10.1609/aaai.v37i12.26732.

[70] Aryaman Gupta, Yusuf Umut Ciftci, and Somil Bansal. "Enhancing Robot Safety via MLLM-Based Semantic Interpretation of Failure Data". In: *Workshop on Out-of-Distribution Generalization in Robotics (OOD@RSS)*. 2025.

[71] Ayush Gupta, Mengyu Zhang, and Saurabh Gupta. "Estimating Perceptual Uncertainty to Predict Robust Motion Plans". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[72] Dan Hendrycks et al. "Scaling Out-of-Distribution Detection for Real-World Settings". In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, pp. 8759–8773.

[73] Kai-Chieh Hsu et al. "Safety and Liveness Guarantees through Reach-Avoid Reinforcement Learning". In: *Proceedings of Robotics: Science and Systems*. Virtual, July 2021. DOI: 10.15607/RSS.2021.XVII.077.

[74] Haimin Hu et al. "Reach-sdp: Reachability analysis of closed-loop systems with neural network controllers via semidefinite programming". In: *2020 59th IEEE conference on decision and control (CDC)*. IEEE. 2020, pp. 5929–5934.

[75] Yafei Hu et al. "Toward General-Purpose Robots via Foundation Models: A Survey and Meta-Analysis". In: *arXiv preprint arXiv:2312.08782* (2023). DOI: 10.48550/arXiv.2312.08782.

[76] Yu Inatsu, Shogo Iwazaki, and Ichiro Takeuchi. "Active learning for distributionally robust level-set estimation". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 4574–4584.

[77] Inigo Incer. "The Algebra of Contracts". PhD thesis. EECS Department, University of California, Berkeley, May 2022. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-99.html.

[78] Inigo Incer et al. "Pacti: Assume-Guarantee Contracts for Efficient Compositional Analysis and Design". In: *ACM Transactions on Cyber-Physical Systems* 9.1 (2025), pp. 1–35. DOI: 10.1145/3704736.

[79] Íñigo Íncer Romeo et al. "Quotient for assume-guarantee contracts". In: *Proceedings of the 16th ACM-IEEE International Conference on Formal Methods and Models for System Design*. MEMOCODE '18. Beijing, China: IEEE Press, 2018, pp. 67–77. ISBN: 9781538661956.

[80] Rolf Isermann. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Berlin, Heidelberg: Springer, 2006. DOI: 10.1007/3-540-30368-5.

[81] Hideaki Ishibashi et al. "An (ϵ, δ)-Accurate Level Set Estimation with a Stopping Criterion". In: *Proceedings of the 41st International Conference on Machine Learning (ICML)*. Proceedings of Machine Learning Research. PMLR, 2024.

[82] Boris Ivanovic and Marco Pavone. "Injecting Planning-Awareness into Prediction and Detection Evaluation". In: *2022 IEEE Intelligent Vehicles Symposium (IV)* (2021), pp. 821–828.

[83] Boris Ivanovic et al. "Propagating State Uncertainty Through Trajectory Forecasting". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2022.

[84] Shogo Iwazaki, Yu Inatsu, and Ichiro Takeuchi. "Bayesian Experimental Design for Finding Reliable Level Set Under Input Uncertainty". In: *IEEE Access* 8 (Jan. 2020), pp. 203982–203993. DOI: 10.1109/ACCESS.2020.3036863.

[85] Hyun Joe Jeong et al. "Parameterized Fast and Safe Tracking (FaSTrack) using DeepReach". In: *Proceedings of the 6th Annual Learning for Dynamics and Control Conference (L4DC)*. Vol. 242. Proceedings of Machine Learning Research. PMLR, 2024, pp. 1006–1017. URL: https://proceedings.mlr.press/v242/jeong24a.html.

[86] H. M. Dipu Kabir et al. "Neural Network-Based Uncertainty Quantification: A Survey of Methodologies and Applications". In: *IEEE Access* 6 (2018), pp. 36218–36234. DOI: 10.1109/ACCESS.2018.2836917.

[87] Peter Karkus et al. "DiffStack: A Differentiable and Modular Control Stack for Autonomous Vehicles". In: *6th Annual Conference on Robot Learning*. 2022.

[88] Sydney M. Katz et al. "Efficient Determination of Safety Requirements for Perception Systems". In: *Digital Avionics Systems Conference*. 2023.

[89] Alex Kendall and Yarin Gal. "What uncertainties do we need in Bayesian deep learning for computer vision?" In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 5580–5590. ISBN: 9781510860964.

[90] Shayan Kiyani et al. "Decision Theoretic Foundations for Conformal Prediction: Optimal Uncertainty Quantification for Risk-Averse Agents". In: *Proceedings of the 42nd International Conference on Machine Learning*. Vol. 267. Proceedings of Machine Learning Research. PMLR, 2025, pp. 30943–30965.

[91] Mykel J. Kochenderfer et al. *Algorithms for Validation*. The MIT Press, 2026. ISBN: 9780262056014.

[92] Roger Koenker. *Quantile regression*. Vol. 38. Cambridge university press, 2005.

[93] Fabian Küppers et al. "Multivariate Confidence Calibration for Object Detection". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2020), pp. 1322–1330.

[94] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6405–6416. ISBN: 9781510860964.

[95] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep Learning". In: *Nature* 521.7553 (May 2015), pp. 436–444. DOI: 10.1038/nature14539.

[96] Jongseo Lee et al. "Trust Your Robots! Predictive Uncertainty Estimation of Neural Networks with Sparse Gaussian Processes". In: *Conference on Robot Learning*. 2021. URL: https://api.semanticscholar.org/CorpusID:237350945.

[97] Jordan Lekeufack et al. "Conformal Decision Theory: Safe Autonomous Decisions from Imperfect Predictions". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 11668–11675. DOI: 10.1109/ICRA57147.2024.10610041.

[98] Thomas Lew and Marco Pavone. "Sampling-based reachability analysis: A random set theory approach with adversarial sampling". In: *Conference on robot learning*. PMLR. 2021, pp. 2055–2070.

[99] Jingqi Li et al. "Certifiable reachability learning using a new lipschitz continuous value function". In: *IEEE Robotics and Automation Letters* 10.4 (2025), pp. 3582–3589.

[100] Yucen Lily Li et al. "Out-of-Distribution Detection Methods Answer the Wrong Questions". In: *arXiv preprint arXiv:2507.01831* (2025). DOI: 10.48550/arXiv.2507.01831.

[101] Shiyu Liang, Yixuan Li, and R. Srikant. "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=H1VGkIxRZ.

[102] Lucas Liebenwein et al. "Sampling-based approximation algorithms for reachability analysis with provable guarantees". In: *Robotics: Sciences and Systems*. 2018.

[103] Albert Lin and Somil Bansal. "Generating formal safety assurances for high-dimensional reachability". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 10525–10531.

[104] Albert Lin and Somil Bansal. "Verification of neural reachable tubes via scenario optimization and conformal prediction". In: *6th Annual Learning for Dynamics & Control Conference*. PMLR. 2024, pp. 719–731.

[105] Albert Lin, Alessandro Pinto, and Somil Bansal. "Robust Verification of Controllers under State Uncertainty via Hamilton-Jacobi Reachability Analysis". In: *Proceedings of The 8th Annual Learning for Dynamics and Control Conference*. Vol. 331. Proceedings of Machine Learning Research. PMLR, 2026, pp. 1432–1446.

[106] Lars Lindemann et al. *Formal Verification and Control with Conformal Prediction: Practical Safety Guarantees for Autonomous Systems*. Foundations and Trends in Systems and Control. Now Publishers, 2023.

[107] Lars Lindemann et al. "Safe planning in dynamic environments using conformal prediction". In: *IEEE Robotics and Automation Letters* (2023).

[108] Weitang Liu et al. "Energy-based out-of-distribution detection". In: NIPS '20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.

[109] Antonio Loquercio, Mattia Segu, and Davide Scaramuzza. "A General Framework for Uncertainty Estimation in Deep Learning". In: *IEEE Robotics and Automation Letters* PP (Feb. 2020), pp. 1–1. DOI: 10.1109/LRA.2020.2974682.

[110] Rachel Luo et al. "Sample-efficient safety assurances using conformal prediction". In: *Algorithmic Foundations of Robotics XV: Proceedings of the Fifteenth Workshop on the Algorithmic Foundations of Robotics*. Springer. 2022, pp. 149–169.

[111] Björn Lütjens, Michael Everett, and Jonathan P. How. "Safe Reinforcement Learning with Model Uncertainty Estimates". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2019.

[112] John Lygeros. "On reachability and minimum cost optimal control". In: *Automatica* 40.6 (2004), pp. 917–927.

[113] Kostas Margellos and John Lygeros. "Hamilton–Jacobi formulation for reach–avoid differential games". In: *IEEE Transactions on automatic control* 56.8 (2011), pp. 1849–1861.

[114] Blake Mason et al. "Nearly Optimal Algorithms for Level Set Estimation". In: *Proceedings of the 40th International Conference on Machine Learning (ICML)*. Vol. 2023. Proceedings of Machine Learning Research. PMLR, 2023, pp. 12345–12356. URL: https://proceedings.mlr.press/v202/mason23a.html.

[115] Nikolai Matni, Aaron D. Ames, and John C. Doyle. "A Quantitative Framework for Layered Multirate Control: Toward a Theory of Control Architecture". In: *IEEE Control Systems* 44 (June 2024), pp. 52–94. ISSN: 1066-033X. DOI: 10.1109/mcs.2024.3382388.

[116] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. "A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games". In: *IEEE Transactions on automatic control* 50.7 (2005), pp. 947–957.

[117] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press, 2012.

[118] Anish Muthali et al. "Multi-Agent Reachability Calibration with Conformal Prediction". In: *2023 IEEE 62nd Conference on Decision and Control (CDC)*. IEEE, 2023, pp. 6596–6603. DOI: 10.1109/CDC49753.2023.10383723.

[119] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. "Obtaining well calibrated probabilities using bayesian binning". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI'15. Austin, Texas: AAAI Press, 2015, pp. 2901–2907. ISBN: 0262511290.

[120] Kensuke Nakamura and Somil Bansal. "Online Update of Safety Assurances Using Confidence-Based Predictions". In: *CoRR* abs/2210.01199 (2022).

[121] Kensuke Nakamura, Lasse Peters, and Andrea Bajcsy. "Generalizing Safety Beyond Collision-Avoidance via Latent-Space Reachability Analysis". In: *International Conference on Learning Representations (ICLR)*. 2026.

[122] Kensuke Nakamura, Thomas Tian, and Andrea V. Bajcsy. "Not All Errors Are Made Equal: A Regret Metric for Detecting System-level Trajectory Prediction Failures". In: *Conference on Robot Learning*. 2024.

[123] Radford M. Neal. *Bayesian Learning for Neural Networks*. New York, NY, USA: Springer, 1996.

[124] Pierluigi Nuzzo et al. "A Platform-Based Design Methodology With Contracts and Related Tools for the Design of Cyber-Physical Systems". In: *Proceedings of the IEEE* 103 (Sept. 2015), pp. 2104–2132. DOI: 10.1109/JPROC.2015.2453253.

[125] Niall O'Sullivan, Licio Romao, and Kostas Margellos. "Bridging Conformal Prediction and Scenario Optimization". In: *2025 IEEE 64th Conference on Decision and Control (CDC)*. IEEE, 2026, pp. 6114–6121. DOI: 10.1109/CDC57313.2025.11312620.

[126] Dario Paccagnan, Marco C. Campi, and Simone Garatti. "The Pick-to-Learn Algorithm: Empowering Compression for Tight Generalization Bounds and Improved Post-training Performance". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023.

[127] Christos H. Papadimitriou and John N. Tsitsiklis. "The Complexity of Markov Decision Processes". In: *Mathematics of Operations Research* 12.3 (Aug. 1987), pp. 441–450. DOI: 10.1287/moor.12.3.441.

[128] Plato. "Apology". In: *Plato: Five Dialogues*. Ed. by G. M. A. Grube and John M. Cooper. Trans. by G. M. A. Grube. 2nd. Indianapolis, IN, USA: Hackett Publishing Company, 2002.

[129] Janis Postels et al. "On the Practicality of Deterministic Epistemic Uncertainty". In: *International Conference on Machine Learning*. 2021.

[130] Janis Postels et al. "Sampling-Free Epistemic Uncertainty Estimation Using Approximated Variance Propagation". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), pp. 2931–2940.

[131] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[132] Tim Salzmann et al. "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data". In: *European Conference on Computer Vision*. 2020.

[133] Alberto Sangiovanni-Vincentelli, Werner Damm, and Roberto Passerone. "Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems". In: *European Journal on Control* (May 2012). URL: http://www.musyc.org/pubs/390.html.

[134] Leonardo Santos et al. "Updating Robot Safety Representations Online from Natural Language Feedback". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 7778–7785. DOI: 10.1109/ICRA55743.2025.11127680.

[135] AQM Sazzad Sayyed et al. "Out-of-Distribution Detection in Computer Vision: A Comprehensive Survey and Research Challenges". In: *ACM Computing Surveys* (2026). DOI: 10.1145/3822404.

[136] Edward Schmerling. *hj_reachability*. https://github.com/StanfordASL/hj_reachability. 2021.

[137] Sanjit A. Seshia, Dorsa Sadigh, and S. Shankar Sastry. "Toward Verified Artificial Intelligence". In: *Communications of the ACM* 65.7 (July 2022), pp. 46–55. DOI: 10.1145/3503914.

[138] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[139] Apoorva Sharma, Navid Azizan, and Marco Pavone. "Sketching Curvature for Efficient Out-of-Distribution Detection for Deep Neural Networks". In: *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*. Vol. 161. Proceedings of Machine Learning Research. PMLR, 2021, pp. 1958–1967.

[140] Rohan Sinha et al. "A System-Level View on Out-of-Distribution Data in Robotics". In: *arXiv preprint arXiv:2212.14020* (2022). DOI: 10.48550/arXiv.2212.14020.

[141] Rohan Sinha et al. "Real-Time Anomaly Detection and Reactive Planning with Large Language Models". In: *Proceedings of Robotics: Science and Systems (RSS)*. Delft, Netherlands, July 2024. DOI: 10.15607/RSS.2024.XX.114.

[142] Ebonye Smith et al. *From Global to Local: Hierarchical Probabilistic Verification for Reachability Learning*. 2026. DOI: 10.48550/arXiv.2603.24990. arXiv: 2603.24990 [eess.SY].

[143] Ingo Steinwart and Andreas Christmann. "Estimating Conditional Quantiles with the Help of the Pinball Loss". In: *Bernoulli* 17.1 (2011), pp. 211–225. DOI: 10.3150/10-BEJ267.

[144] Yiyou Sun et al. "Out-of-Distribution Detection with Deep Nearest Neighbors". In: *Proceedings of the 39th International Conference on Machine Learning*. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, pp. 20827–20840.

[145] Shion Takeno et al. "Regret Analysis of Posterior Sampling-Based Expected Improvement for Bayesian Optimization". In: *Transactions on Machine Learning Research* (2025). DOI: 10.48550/arXiv.2507.09828.

[146] Nassim Nicholas Taleb. *The Black Swan: The Impact of the Highly Improbable*. New York: Random House, 2007.

[147] Zhi Tian et al. "FCOS: Fully Convolutional One-Stage Object Detection". In: Oct. 2019, pp. 9626–9635. DOI: 10.1109/ICCV.2019.00972.

[148] Ryan J Tibshirani et al. "Conformal prediction under covariate shift". In: *Advances in Neural Information Processing Systems* 32 (2019).

[149] Claire Tomlin, George J Pappas, and Shankar Sastry. "Conflict resolution for air traffic management: A study in multiagent hybrid systems". In: *IEEE Transactions on automatic control* 43.4 (2002), pp. 509–521.

[150] Hazem Torfah et al. "Learning Monitorable Operational Design Domains for Assured Autonomy". In: *Proceedings of the International Symposium on Automated Technology for Verification and Analysis (ATVA)*. Oct. 2022.

[151] Renukanandan Tumu et al. "Physics Constrained Motion Prediction with Uncertainty Quantification". In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2023, pp. 1–8. DOI: 10.1109/IV55152.2023.10186812.

[152] Balakrishnan Varadarajan et al. "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction". In: *International Conference on Robotics and Automation*. IEEE. 2022, pp. 7814–7821.

[153] Eugene Vinitsky et al. "Nocturne: A Scalable Driving Benchmark for Bringing Multi-Agent Learning One Step Closer to the Real World". In: *Advances in Neural Information Processing Systems*. Vol. 35. Datasets and Benchmarks Track. 2022.

[154] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Vol. 29. Springer, 2005.

[155] Kim P Wabersich et al. "Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems". In: *IEEE Control Systems Magazine* 43.5 (2023), pp. 137–177.

[156] A. Wald. *Sequential Analysis*. John Wiley and Sons, New York, 1947.

[157] Lirui Wang et al. "GenSim: Generating Robotic Simulation Tasks via Large Language Models". In: *International Conference on Learning Representations (ICLR)*. 2024.

[158] Xinshuo Weng et al. "PARA-Drive: Parallelized Architecture for Real-Time Autonomous Driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 15148–15157. DOI: 10.1109/CVPR52733.2024.01463.

[159] Grady Williams et al. "Aggressive driving with model predictive path integral control". In: *IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016, pp. 1433–1440.

[160] Timothy Williamson. *Knowledge and Its Limits*. Oxford, UK: Oxford University Press, 2000.

[161] Jiancong Xiao et al. "Understanding Adversarial Robustness Against On-Manifold Adversarial Examples". In: *Pattern Recognition* 159 (2025), p. 111071. DOI: 10.1016/j.patcog.2024.111071.

[162] Chen Xu and Yao Xie. "Conformal prediction interval for dynamic time-series". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 11559–11569.

[163] Beyazit Yalcinkaya et al. "Compositional Simulation-Based Analysis of AI-Based Autonomous Systems for Markovian Specifications". In: *23rd International Conference on Runtime Verification (RV)*. Ed. by Panagiotis Katsaros and Laura Nenzi. Vol. 14245. Lecture Notes in Computer Science. Springer, 2023, pp. 191–212.

[164] Beyazit Yalcinkaya et al. "Ulgen: A Runtime Assurance Framework for Programming Safe Cyber-Physical Systems". In: *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems* 42.11 (2023), pp. 3679–3692.

[165] Fengjun Yang and Nikolai Matni. "Coordinating Planning and Tracking in Layered Control Policies via Actor-Critic Learning". In: *2024 IEEE 63rd Conference on Decision and Control (CDC)*. IEEE, 2024, pp. 3762–3769. DOI: 10.1109/CDC56724.2024.10886068.

[166] Jiayu Yao et al. "Quality of Uncertainty Quantification for Bayesian Neural Network Inference". In: *ICML Workshop on Uncertainty and Robustness in Deep Learning (UDL)*. 2019.

[167] Christopher Yeh et al. "End-to-End Conformal Calibration for Optimization Under Uncertainty". In: *Transactions on Machine Learning Research* (2025). URL: https://openreview.net/forum?id=yM8qkT0f9H.

[168] Andrea Zanette, Junzi Zhang, and Mykel J. Kochenderfer. "Robust Super-Level Set Estimation using Gaussian Processes". In: *ECML/PKDD*. 2018.

[169] Gioele Zardini. "Co-Design of Complex Systems: From Autonomy to Future Mobility Systems". PhD thesis. ETH Zurich, 2023.

[170] Fengxue Zhang and Yuxin Chen. "Direct Regret Optimization in Bayesian Optimization". In: *CoRR* abs/2507.06529 (2025). DOI: 10.48550/arXiv.2507.06529. URL: https://arxiv.org/abs/2507.06529.

[171] Hanli Zhang et al. "Why Change Your Controller When You Can Change Your Planner: Drag-Aware Trajectory Generation for Quadrotor Systems". In: *arXiv preprint 2401.04960* (2024).